ISSN 0868 - 6157

Совместное советско-американское предприятие «СОВАМИНКО»

КОМПЬЮТЕР ПРЕСС

ОБОЗРЕНИЕ ЗАРУБЕЖНОЙ ПРЕССЫ

Практическое программирование на dBASE

Программирование для "мыши"

СУБД нового поколения

"Информатика-90"



Совместное советско-американское предприятие «СОВАМИНКО»

КОМПЬЮТЕР ПРЕСС

ОБОЗРЕНИЕ ЗАРУБЕЖНОЙ ПРЕССЫ

СОДЕРЖАНИЕ

11'90

Совместное советско-американское предприятие «СОВАМИНКО»
КОМПЬЮТЕР ПРЕСС
ОБОЗРЕНИЕ ЗАРУБЕЖНОЙ ПРЕССЫ

Адрес редакции:
113093, г.Москва, аб.ящик 37
Тел. для справок: 150-17-03
Бюро рекламы: 156-81-33
Факс: 200-22-89

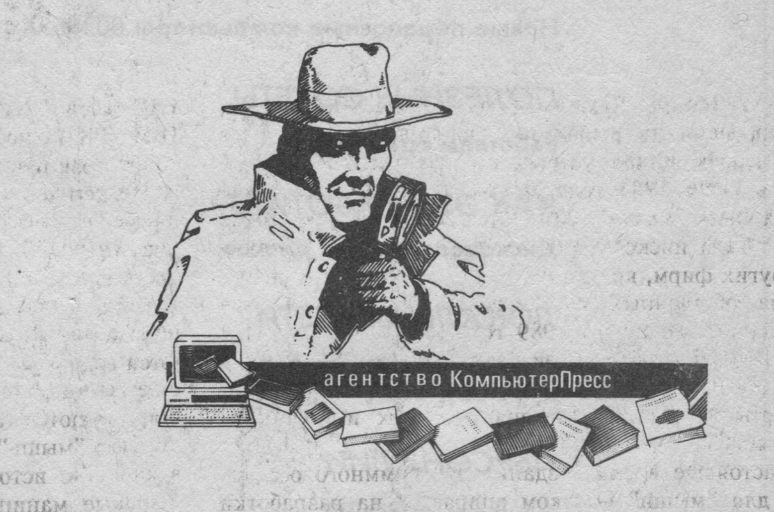
Внимание !

Подписка на журнал «КомпьютерПресс»

У Вас есть желание приобрести вычислительную технику, но Вы не знаете как это сделать? А может быть у Вас уже есть компьютер, но Вы не знаете, какие программы помогут Вам оживить его? Как определить, что именно больше всего подходит для Вашей организации? Или эти задачи для Вас уже в прошлом и Вас интересует передовой опыт зарубежных коллег в области аппаратного и программного обеспечения?

Решить все эти проблемы поможет обозрение «КомпьютерПресс» — единственный в Советском Союзе ежемесячный журнал по проблемам информатики и компьютерных технологий.

Условия подписки на «КомпьютерПресс» Вы найдете на странице 79.

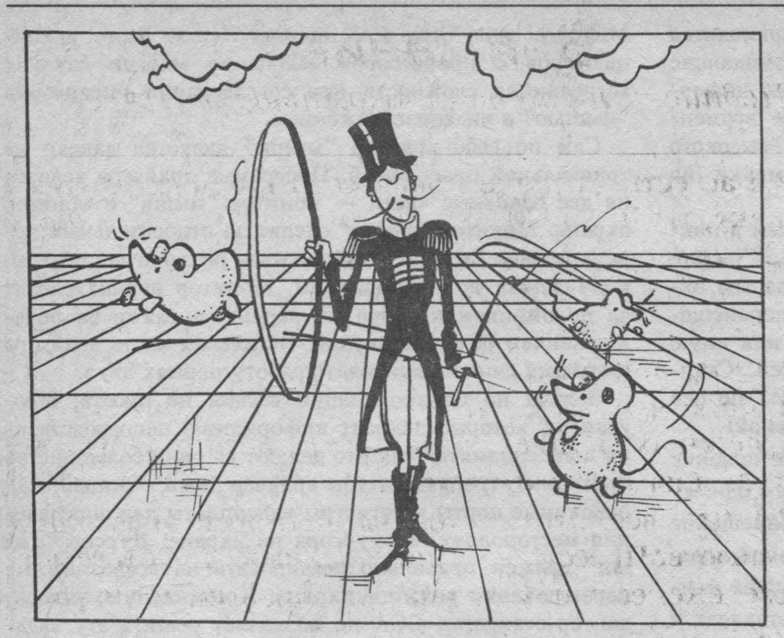


Годовая подписка на наш журнал —

это экономия вашего времени!

Сдано в набор 21.11.90. Подписано к печати 23.11.90. Формат 84x108/16. Печать офсетная.
Усл.печ.л.8,4+0,32 (обл.). Тираж 100000 экз.(1 завод—55 000). Заказ 1319 Цена 2 р. 80 к.

Типография издательства «Калининградская правда»
236000, г.Калининград, ул.Карла Маркса, 18



В отечественной литературе практически отсутствует информация по программированию драйвера мыши. Эта статья призвана восполнить этот пробел. В ней описываются функции драйвера мыши фирмы Microsoft и рассматриваются вопросы использования этих функций применительно к различным режимам работы видеоадаптера.

Программирование для "мыши"

Фирма Mouse Systems была первой компанией, представившей на рынок персональных компьютеров такое периферийное устройство, как "мышь". После этого, в июне 1983 года, фирма Microsoft выпустила свой вариант "мыши". Хотя продукция фирмы Mouse Systems была нисколько не хуже аналогичной продукции других фирм, компания не разрабатывала и не выпускала собственных рекомендаций по программированию "мыши" до конца 1989 года. Фирмы Microsoft и Logitech опубликовали свои разработки в этой области несколькими годами раньше, определив тем самым стандарты в производстве программных интерфейсов для "мыши".

В настоящее время создание программного обеспечения для "мыши" целиком опирается на разработки фирмы Microsoft с использованием прерываний центрального процессора и набора функций (вызываемых при помощи прерываний), служащих для обработки данных, полученных от позиционирующего устройства.

Первые "мыши" фирмы Microsoft имели специальную плату, устанавливаемую на системную шину компьютера, к которой подключалась сама "мышь". В последующих моделях и их модификациях для подключения "мыши" использовался уже просто последовательный порт, хотя для современных компьютеров разработаны по меньшей мере два типа специальных интерфейсов "мыши" — Microsoft InPort интерфейс и IBM PS/2 последовательный "pointing device port" (порт позиционирующего устройства).

На сегодняшний день на рынке персональных компьютеров имеется большое количество различных типов "мышей". Они отличаются в основном разными способами преобразования физического перемещения "мыши" в последовательность электрических сигналов, передаваемых в компьютер. Для этой цели используются всевозможные колесики, шарики и светоотражающие решетки (дифракционные решетки). Помимо "мыши", имеются также и другие устройства, которые эмулируют "мышь" программными средствами, используя в качестве источника сигналов различные джойстики, шаровые манипуляторы (trackballs), сенсорные подушечки (touch pads), как например "Koala pad". Джойстики и шаровые манипуляторы оказываются полезными, если рабочее пространство на вашем столе сильно ограничено. Большинство таких устройств подключается к системе через Microsoft Mouse API (интерфейс прикладных программ для "мыши" фирмы Microsoft).

В качестве единицы измерения физического перемещения "мыши" принято использовать "микки" (mickey). Этот термин был введен Биллом Гейтсом (Bill Gates), президентом фирмы Microsoft, воспользовавшимся именем известного мультипликационного персонажа Микки Мауса (Mickey Mouse). Обычно на

1 дюйм (2.54 см) перемещения "мыши" приходится около 200 микки. Мышь подсчитывает поступающие сигналы микки и затем передает накопленную информацию своему драйверу через определенные временные интервалы. Некоторые новые "мыши" "высокого разрешения" вырабатывают 320 или 400 микки на дюйм.

Драйвер "мыши" преобразует сигналы микки в пикселы на экране (пиксел — pixel или PEL, "picture element" — соответствует одному элементу растра экрана). Количество микки, необходимое для перемещения курсора на экране на один пиксел в ту или иную сторону, регулируется специальной функцией. Стандартное соотношение микки-пиксел — это 1:1 по оси X (горизонтальной) и 2:1 по оси Y (вертикальной).

В графических режимах минимальное перемещение курсора "мыши" на экране может быть равным одному пикселу; в текстовых режимах единичное перемещение курсора "мыши" обычно равно одной символьной клетке. Для текстового режима в стандарте Hercules наименьший шаг движения курсора "мыши" составляет 9 пикселов по горизонтали и 14 по вертикали.

Когда вы передвигаете "мышь" по столу, пропорционально с ней движется и курсор на экране. Для того, чтобы иметь возможность легко перемещать курсор на экране в широких пределах, нужно установить значение соотношения движений "мыши" и курсора небольшим. При этом, однако, возникнут трудности с точной установкой курсора на нужную позицию на экране. Для разрешения этой проблемы некоторые простые драйверы "мыши" используют так называемый "порог удвоенной скорости". Мышь и курсор перемещаются с соотношением скоростей 1:1 до тех пор, пока не будет достигнут определенный порог скорости движения "мыши" (измеряемой в микки в секунду). Затем драйвер начинает удваивать количество принимаемых сигналов микки перед их последующей обработкой. Тем самым достигается эффективное увеличение скорости движения курсора на экране. Драйверы "мыши" с удвоением скорости являются наиболее распространенными.

Наилучшие результаты получаются при использовании так называемых "баллистических" драйверов. Такой драйвер следит за скоростью поступления сигналов микки и преобразует их количество в соответствии с некоторой арифметической функцией или таблицей. Таким образом достигается плавное изменение соотношения микки-пиксел и, соответственно, скорости движения курсора на экране от более медленной к более быстрой и наоборот.

Драйвер "мыши" тесно связан с обработкой видеоизображения. Пользователи, работающие с теми пакетами, в которых интенсивно используется "мышь", должны хорошо представлять себе сложности, возникающие при программировании "мыши". Наибольшие трудности возникают при разработке графических программ и программ по обработке изображений, подобных пакетам PC Paintbrush фирмы Zsoft, Designer фирмы Micrograf или Gray F/X фирмы Xerox. Даже у компьютеров с процессором 80386 во многих случаях встречаются сложности при согласовании управления "мышью" и видеоизображением.

Сам по себе драйвер "мыши" является далеко не тривиальной программой. Программа драйвера делится на две основные части — монитор "мыши" и монитор экрана. Монитор "мыши" следит за относительным перемещением "мыши", подсчитывая различные сигналы, получаемые по прерываниям. Монитор экрана следит за положением курсора на экране, а также за переключением режимов экрана. Остальная часть драйвера старается синхронизировать работу первых двух.

Задача по синхронизации весьма не проста. Программы, которые заносят информацию непосредственно в видеопамять (как это делают сегодня большинство программ), требуют, чтобы драйвер "мыши" опрашивал различные порты и регистры видеоплаты для определения местоположения курсора на экране. Курсор "мыши" должен правильно реагировать на всевозможные переключения режимов экрана. К сожалению, видеоадаптер стандарта EGA не позволяет решить эту задачу. Большинство регистров EGA доступны только для записи, что зачастую не дает возможности драйверу "мыши" определить не только положение курсора на экране, но даже и текущий режим самого экрана.

Решение этой проблемы осуществляется при помощи использования так называемой "библиотеки регистрового интерфейса" EGA-Register Interface Library (RIL), являющейся составной частью драйвера "мыши" фирмы Microsoft. Программное обеспечение RIL осуществляет запись видеоинформации в библиотеку RIL, которая затем передает ее на видеоплату. Драйвер "мыши" в дальнейшем обращается также к RIL вместо видеоплаты. Хотя большинство драйверов различных фирм умеют определять наличие RIL и используют ее, не все из них предоставляют само программное обеспечение RIL.

Существуют также программы — такие как Windows, AutoCAD или First Publisher — которые работают в режимах EGA и не имеют проблем с согласованием "мыши" и экрана. В этих программах имеется встроенная поддержка "мыши", и драйвер "мыши" всегда точно знает обо всех переключениях режимов экрана (внешний драйвер "мыши", например, фирмы Microsoft, при этом не используется).

ОПИСАНИЕ РАБОТЫ ДРАЙВЕРА

Использование регистров

Обращения к функциям драйвера "мыши" осуществляются аналогично обычным обращениям к функциям DOS. В регистры центрального процессора заносятся необходимые значения, и далее производится вызов прерывания 33h (прерывание обслуживания драйвера "мыши"). Программа обслуживания прерывания выполняет требуемые действия и, если это предусмотрено, возвращает значения полученных результатов.

Драйвер "мыши" фирмы Microsoft также "перехватывает" вектор прерывания 10h BIOS (базовой системы ввода-вывода) и отслеживает переключения режимов экрана. Драйвер автоматически настраивается на любой из поддерживаемых BIOS видеорежимов. Фирма Microsoft предлагает драйвер, поддерживающий 35 прикладных функций, у драйверов других фирм количество выполняемых функций может быть иным. Драйвер "мыши" не производит проверки правильности начальных значений, вводимых в функцию, об этом должна заботиться прикладная программа. В случае обращения к непредусмотренной функции большинство драйверов просто игнорируют вызов. По общепринятому соглашению номера функций драйвера "мыши" даются в десятичных числах, в отличие от обычно используемого для таких целей шестнадцатиричного формата.

Графические и текстовые курсоры

Драйвер "мыши" поддерживает работу с тремя разновидностями курсора — жестким (аппаратным) текстовым курсором, мягким (программируемым) текстовым курсором и графическим курсором. Жесткий текстовый курсор — это мигающий курсор, поддерживаемый видеоадаптером, который может перемещаться только по символьным клеткам на экране. Этот курсор может иметь форму прямоугольного блока или подчеркивающей линии. Мягкий текстовый курсор поддерживается программными средствами драйвера. Драйвер изменяет в экранной памяти атрибуты текущего символа, меняя таким образом цвет фона, цвет символа или и то и другое одновременно. Такой курсор также перемещается только по символьным клеткам экрана. Графический курсор имеет вид некоторого графического объекта, перемещающегося поверх изображения на экране.

Вы можете выбрать любой из трех перечисленных типов курсора для использования в соответствующем режиме экрана, однако только один курсор может присутствовать на экране в текущий момент времени. Внутри вашей прикладной программы вы можете также производить переключения между различными типами курсора.

Формирование графического курсора

Графический курсор может появляться на экране или исчезать с экрана при помощи вызова соответствующих функций. Этот курсор определяется некоторым блоком пикселов, имеющим прямоугольную форму. Когда такой блок перемещается по экрану и взаимодействует с пикселами изображения, находящегося "под" ним, происходит формирование фона и изображения графического курсора. Результаты этого взаимодействия определяются содержимым двух массивов, размером 16 на 16 бит каждый, один из которых является маской экрана, а другой — маской курсора. Маска экрана определяет, какая часть пикселов графического блока курсора будет формировать образ курсора, а какая часть будет фоном курсора. Маска курсора определяет те пикселы, которые будут участвовать в формировании цвета курсора. Перед тем как произвести какие-либо изменения изображения на том участке экрана, который находится непосредственно под курсором, курсор необходимо спрятать (восстановить изображение на месте курсора), для того, чтобы впоследствии, при перемещении курсора, на новое изображение не наложилась часть старого.

Необходимо иметь в виду, что в режиме высокого разрешения (EGA, VGA) графический блок курсора имеет размеры 16 на 16 пикселов; в режиме среднего разрешения (CGA, 4 цвета) — 8 на 16 пикселов; в режиме среднего разрешения (EGA, VGA, 640x200, 16 цветов) — 4 на 16 пикселов.

При формировании графического курсора драйвер использует данные, содержащиеся в экранной памяти компьютера и определяющие цвета каждого пиксела на экране. Для каждого пиксела графического блока курсора драйвер выполняет следующие побитовые логические операции: логическое умножение (AND) маски экрана на поле экрана, лежащее под курсором, и далее логическую операцию исключающего ИЛИ (XOR) маски экрана и результата предыдущей операции.

В таблице представлены варианты возможных сочетаний:

Бит маски экрана	Бит маски курсора	Результирующий бит на экране
0	0	0
0	1	1
1	0	не изменяется
1	1	инвертируется

Для каждой поддерживаемой драйвером функции "мыши" под положением графического курсора на экране подразумеваются координаты некоторой точки экрана, находящейся под блоком курсора и носящей название "горячего пятна" курсора. Обычно (по умолчанию) "горячим пятном" курсора является левый верхний угол графического блока курсора с относительными координатами (0,0), но по желанию в качестве "горячего пятна" можно выбрать любую другую точку курсора.

Мягкий текстовый курсор

Этот текстовый курсор вы можете использовать во время работы вашего компьютера в одном из текстовых режимов. Изменяя атрибуты символа под курсором, можно получать различные сочетания цветов этого символа и его фона, выделяя таким образом нужную текстовую клетку.

Желаемый результат достигается путем задания некоторых значений для двух 16-ти битных масок — маски экрана и маски курсора (аналогично графическому курсору) — которые впоследствии будут переда-

ны драйверу. Каждая из этих масок содержит в себе следующие битовые поля: бит 15 устанавливает состояние мигающего (1) или немигающего (0) символа; биты 12-14 определяют цвет фона данной текстовой клетки; биты 8-10 определяют цвет символа; биты 0-7 содержат ASCII-код символа.

Значениями маски экрана и маски курсора определяются новые атрибуты символа, находящегося под курсором. Маска экрана определяет, какие из атрибутов символа смогут быть далее установлены. Маска курсора определяет, каким образом изменятся эти атрибуты.

Драйвер обращается непосредственно к текстовой памяти экрана, выбирая из нее данные по интересующему символу. Далее производится логическое умножение (AND) выбранных с экрана данных на маску экрана, и затем операция исключающего ИЛИ (XOR) между результатом предыдущей операции и маской курсора.

Под положением текстового курсора на экране драйвер подразумевает координаты символа, находящегося под курсором.

Жесткий текстовый курсор

Этот курсор, поддерживаемый аппаратными средствами видеоадаптера, также может быть использован при работе компьютера в одном из текстовых режимов. Ширина курсора равна ширине одной текстовой клетки, а его высота регулируется программными средствами в пределах высоты текстовой клетки. Высота курсора определяется количеством скан-линий (отрезков строк горизонтальной развертки), которые будут участвовать в формировании курсора. Количество скан-линий, составляющих одну текстовую клетку экрана, зависит от типа используемого дисплея и от режима экрана. Верхняя скан-линия текстовой клетки имеет номер 0.

Клавиши "мыши"

Функции драйвера "мыши" позволяют определять текущее состояние клавиш "мыши", а также подсчитывать, сколько раз определенная клавиша была нажата или отпущена за известный промежуток времени. Текущее состояние клавиш определяется байтом состояния, возвращаемым соответствующей функцией, где:

Бит 0 — Состояние левой клавиши
Бит 1 — Состояние правой клавиши
Бит 2 — Состояние средней клавиши (для "мышей" фирмы Mouse Systems)

Если соответствующий бит установлен (1), то клавиша в данный момент нажата; если бит сброшен (0) — то клавиша отпущена.

Каждое нажатие и отпускание клавиши регистрируется в специальном счетчике. Драйвер каждый раз сбрасывает этот счетчик в 0 после того, как его показания будут прочитаны соответствующей функцией, а также при начальной установке.

Внутренний флажок курсора

Драйвер "мыши" постоянно поддерживает так называемый "внутренний флажок курсора". Этот флажок определяет, когда курсор должен появляться на экране. Если флажок равен 0, то курсор становится видимым; если флажок отличен от нуля (принимает любое другое значение), курсор исчезает с экрана.

Вы можете производить вызов функций 1 и 2 сколько угодно раз, однако, каждый вызов функции 2 (запрещающей появление курсора), требует последующего вызова функции 1 (разрешающей появление курсора). Это необходимо для восстановления предыдущего состояния флажка курсора, так как он накапливает количество вызовов функций.

Вызов функций драйвера "мыши" из ассемблерной программы

Пример:

```
; установить графический курсор в точку
; с координатами (150,100)
        mov     ax,4      ; вызов функции 4
        mov     cx,150    ; горизонтальная координата 150
        mov     dx,100    ; вертикальная координата 100
        int     33h       ; обращение к прерыванию 33h
```

Важно иметь в виду, что перед первым обращением к прерыванию 33h нужно проверить наличие драйвера "мыши" в памяти вашего компьютера. Если драйвер не загружен, выполнение прерывания 33h приведет к непредсказуемым результатам. Удостовериться в наличии загруженного драйвера "мыши" можно по следующим признакам:

1) вектор прерывания 33h не равен нулю;
2) вектор прерывания 33h не указывает на инструкцию IRET.

Функции драйвера "мыши"

Функция 0: Начальная установка драйвера и чтение текущего состояния

Функция 0 производит начальную установку и возвращает информацию о текущем состоянии аппаратных и программных средств "мыши". Функция определяет текущий режим экрана, прячет курсор и помещает его в центр экрана, а также задает начальные значения внутренним переменным драйвера в соответствии со следующей таблицей:

Переменная	Значение
внутренний флажок курсора	0FFFFh (-1), курсор спрятан
форма графического курсора	горизонтальный овал
текстовый курсор	обращенное видеоизображение символа
маски, определяемые пользователем	все нули
режим эмуляции светового пера	включен
вертикальное соотношение микки/пиксел	16/8

горизонтальное соотношение микки/пиксел	8/8
минимальные координаты курсора по вертикали и по горизонтали	0,0
максимальные координаты курсора по вертикал и по горизонтали	соответствующие максимальные значения координат экрана в текущем режиме минус единица

Регистры 80i86

Входные значения:	AX	0000h
Возвращаемые значения:	AX	состояние драйвера
		0000h драйвер или аппаратная поддержка не установлены
		FFFFh(-1) начальная установка прошла успешно
	BX	количество клавиш "мыши"
		0000h не две клавиши
		0002h две клавиши
		0003h три клавиши (мышь Mouse Systems)

<u>Функция 1:</u> Сделать курсор видимым

Функция 1 увеличивает на 1 значение счетчика внутреннего флажка курсора. Если флажок равен нулю, курсор становится видимым и появляется на экране.

Начальное значение флажка равно -1, что соответствует спрятанному курсору. В случае, если внутренний флажок уже равен нулю, вызов функции 1 не приводит ни к каким результатам.

Регистры 80i86

Входные значения:	AX	0001h
Возвращаемые значения:	нет	

<u>Функция 2:</u> Сделать курсор невидимым

Функция 2 делает курсор невидимым и уменьшает на 1 значение внутреннего флажка курсора. Несмотря на то, что курсор не виден на экране, он продолжает отслеживать движение "мыши".

Эту функцию рекомендуется вызывать каждый раз перед тем, как произвести какие-либо изменения на том участке экрана, где находится курсор. Этим вы избежите проблем, возникающих из-за взаимодействия курсора с данными на экране.

Нужно иметь в виду, что на каждый вызов функции 2 впоследствии должен быть произведен соответствующий вызов функции 1, для того, чтобы восстановить значение внутреннего флажка курсора. Кроме того, при каждом переключении режима экрана функция 2 вызывается автоматически.

Производите вызов функции 2 в конце вашей программы, чтобы спрятать курсор. Это позволит вам быть уверенным в том, что ничего лишнего не останется на экране по завершении работы программы.

Регистры 80i86

Входные значения:	AX	0002h
Возвращаемые значения:	нет	

<u>Функция 3:</u> Определение местоположения курсора и состояния клавиш "мыши"

Функция 3 возвращает данные по текущему состоянию клавиш "мыши" и положению курсора на экране.

Регистры 80i86

Входные значения:	AX	0003h
Возвращаемые значения:	BX	байт состояния клавиш (BL)
	Бит 0	левая клавиша
	Бит 1	правая клавиша
	Бит 2	средняя клавиша (мышь Mouse Systems)
	Биты 3-7 не используются	
	CX	горизонтальная координата курсора
	DX	вертикальная координата курсора

Если бит установлен (1), то клавиша нажата, если сброшен (0) — клавиша отпущена.

<u>Функция 4:</u> Установить курсор на экране в заданную позицию

Функция 4 устанавливает курсор на экране в заданную позицию. Входные значения координат должны быть в пределах диапазона, установленного драйвером для текущего виртуального экрана "мыши".

Некоторые драйверы "мыши" округляют значение счетчика микки для получения действительной координаты пиксела, другие просто отбрасывают остаток при масштабировании. Это может приводить к некоторой разнице в позиционировании курсора при использовании разных драйверов "мыши", что становится особенно заметным при сочетании новых "мышей" высокого разрешения с дисплеями высокого разрешения.

Регистры 80i86

Входные значения:	AX	0004h
		CX горизонтальная координата курсора
	DX	вертикальная координата курсора
Возвращаемые значения:		нет

<u>Функция 5:</u> Получить информацию о количестве нажатий на клавишу

Функция 5 возвращает информацию о текущем состоянии клавиш, количестве нажатий на указанную клавишу с момента последнего вызова данной функции, а также о позиции курсора в момент последнего нажатия указанной клавиши.

Диапазон счетчика нажатий клавиши колеблется от 0 до 0EFFFh. Индикатор переполнения отсутствует. После вызова данной функции счетчик обнуляется.

Регистры 80i86

Входные значения:	AX	0005h	
	BX	байт идентификации клавиш (BL)	
	Бит 0	левая клавиша	
	Бит 1	правая клавиша	
	Бит 2	средняя клавиша ("мышь" Mouse Systems)	
Возвращаемые значения:	AX	байт состояния клавиш (AL)	
		Бит 0	левая клавиша
		Бит 1	правая клавиша
		Бит 2	средняя клавиша ("мышь" Mouse Systems)
	BX	число нажатий указанной клавиши с момента последнего вызова функции	
	CX	горизонтальная координата курсора в момент нажатия указанной клавиши	
	DX	вертикальная координата курсора в момент нажатия указанной клавиши	

Если бит установлен (1), то клавиша нажата, если сброшен (0) — клавиша отпущена.

Функция 6: Получить информацию о количестве отпусканий клавиши

Функция 6 аналогично функции 5 возвращает информацию о текущем состоянии клавиш, количестве отпусканий указанной клавиши с момента последнего вызова данной функции, а также о позиции курсора в момент последнего отпускания указанной клавиши.

Диапазон счетчика отпусканий клавиши колеблется от 0 до 0EFFFh. Индикатор переполнения отсутствует. После вызова данной функции счетчик обнуляется.

Регистры 80i86

Входные значения:	AX	0006h
	BX	байт идентификации клавиш (BL)
	Бит 0	левая клавиша
	Бит 1	правая клавиша
	Бит 2	средняя клавиша (мышь Mouse Systems)
Возвращаемые значения:	AX	байт состояния клавиш (AL)
	Бит 0	левая клавиша
	Бит 1	правая клавиша
	Бит 2	средняя клавиша (мышь Mouse Systems)
	BX	число отпусканий указанной клавиши с момента последнего вызова функции
	CX	горизонтальная координата курсора в момент отпускания указанной клавиши
	DX	вертикальная координата курсора в момент отпускания указанной клавиши

Если бит установлен (1), то клавиша нажата, если сброшен (0) — клавиша отпущена.

Функция 7: Установить диапазон перемещения курсора по горизонтали (X)

Функция 7 устанавливает горизонтальный диапазон перемещения курсора на экране. В результате текущая горизонтальная координата курсора приводится к новому масштабу. Если в момент вызова горизонтальная координата курсора находилась вне указанного диапазона, курсор помещается в соответствующий край диапазона.

Регистры 80i86

Входные значения:	AX	0007h
	CX	минимальная горизонтальная координата курсора
	DX	максимальная горизонтальная координата курсора
Возвращаемые значения:	нет	

Если минимальное значение больше максимального, функция производит обмен значений.

Функция 8: Установить диапазон перемещения курсора по вертикали (Y)

Функция 8 устанавливает вертикальный диапазон перемещения курсора на экране. В результате текущая вертикальная координата курсора приводится к новому масштабу. Если в момент вызова вертикальная координата курсора находилась вне указанного диапазона, курсор помещается в соответствующий край диапазона.

Регистры 80i86

Входные значения:	AX	0008h
	CX	минимальная вертикальная координата курсора
	DX	максимальная вертикальная координата курсора
Возвращаемые значения:	нет	

Если минимальное значение больше максимального, функция производит обмен значений.

Функция 9: Задать параметры графического курсора

Функция 9 устанавливает цвет, форму и задает координаты "горячего пятна" графического курсора. Как уже было сказано выше (см. примечание к ф.9), этот курсор формируется из графического блока размером 16 на 16 пикселов и определяется двумя массивами 16 на 16 бит каждый (маской экрана и маской курсора). Координаты горячего пятна курсора должны находиться в диапазоне от 0 до 16.

Регистры 80i86

Входные значения:	AX	0009h
	BX	горизонтальная координата горячего пятна курсора в маске курсора
	CX	вертикальная координата горячего пятна курсора в маске курсора
	ES:DX	указатель на массивы масок
	16 слов — маска экрана	
	16 слов — маска курсора	
Возвращаемые значения:	нет	

Каждое слово задает значения 16 пикселов в соответствующем ряду. Младший бит соответствует крайнему правому пикселу.

Функция 10: Задать параметры текстового курсора

Функция 10 осуществляет выбор жесткого или мягкого текстового курсора. Для жесткого текстового курсора задаются первая и последняя скан-линии, принимающие участие в формировании курсора. Для мягкого текстового курсора задаются маски экрана и курсора.

Регистры 80i86

Входные значения:	AX	000Ah
	BX	выбор типа курсора
	00h	мягкий текстовый курсор
	01h	жесткий текстовый курсор
	CX	маска экрана или номер первой скан-линии
	DX	маска курсора или номер последней скан-линии
Возвращаемые значения:	нет	

Функция 11: Прочитать значение счетчика сигналов микки

Функция 11 возвращает число сигналов микки (минимальных приращений перемещения "мыши", регистрируемых аппаратными средствами), накопленное счетчиком с момента последнего вызова данной функции. Положительные значения соответствуют движению "мыши" вправо или вверх. Значения счетчика расположены в интервале от -32768 до 32767. Индикатор переполнения отсутствует. После вызова функции счетчик обнуляется.

Регистры 80i86

Входные значения:	AX	000Bh
Возвращаемые значения:	CX	число микки по горизонтали
	DX	число микки по вертикали

Функция 12: Задать адрес подпрограммы обработки прерывания

Функция 12 передает драйверу адрес входа в подпрограмму обработки прерывания, вызванного некоторым событием. В случае возникновения такого события выполнение прикладной программы временно прерывается, и управление передается по установленному адресу. После завершения работы подпрограммы работа прикладной программы возобновляется в той точке, где была прервана.

Битовая маска вызова определяет условия, вызывающие прерывание. Установленный бит (1) разрешает прерывание, сброшенный (0) — запрещает.

Регистры 80i86

Входные значения:	AX	000Ch
	CX	маска вызова
		бит 0 изменение позиции курсора *
		бит 1 нажата левая клавиша
		бит 2 отпущена левая клавиша
		бит 3 нажата правая клавиша
		бит 4 отпущена правая клавиша
		бит 5 нажата средняя клавиша (Mouse Systems)
		бит 6 отпущена средняя клавиша (Mouse Systems)
		7-15 не используются
	ES:DX	FAR адрес подпрограммы обработки прерывания
Возвращаемые значения:	не определены	

* Драйвер фирмы Microsoft следит за позицией "мыши". Драйверы фирм Logitech и Mouse Systems следят за позицией курсора.

Во время вызова подпрограммы обработки прерывания драйвер передает ей следующие параметры:

AH	маска условия (биты установлены так же, как в маске вызова)
BX	состояние клавиш
CX	горизонтальная координата курсора
DX	вертикальная координата курсора
DI	значение вертикального счетчика микки
SI	значение горизонтального счетчика микки

При завершении работы вашей прикладной программы установите все биты маски вызова равными ну-

лю и вызовите функцию 12. (При выходе из программы оставляйте систему в том же состоянии, что и при входе.)

Функция 13: Включение режима эмуляции светового пера

Функция 13 позволяет "мыши" работать в режиме светового пера. В этом случае обращение к функции светового пера будет возвращать координаты курсора на момент последнего состояния "перо опущено".

Состояния "перо поднято" и "перо опущено" определяются клавишами "мыши": все клавиши отпущены — "перо поднято"; одна клавиша нажата — "перо опущено".

При начальной установке драйвера режим эмуляции светового пера включается по умолчанию. Если в системе присутствует настоящее световое перо, должна быть вызвана функция 14 для запрещения эмуляции.

Регистры 80i86

Входные значения:	AX	000Dh
Возвращаемые значения:	нет	

Функция 14: Запрещение режима эмуляции светового пера

Функция 14 выключает режим эмуляции светового пера. При этом любое последующее обращение к функции светового пера будет возвращать информацию только о состоянии настоящего светового пера.

Регистры 80i86

Входные значения:	AX	000Eh
Возвращаемые значения:	нет	

Функция 15: Установить соотношение микки/пиксел

Соотношение определяется числом микки, приходящимся на 8 пикселов экрана. Значения микки должны быть в пределах от 1 до 0EFFFh. При начальной установке драйвера по умолчанию выбираются следующие соотношения микки/пиксел: по горизонтали — 8/8; по вертикали — 16/8.

Регистры 80i86

Входные значения:	AX	000Fh
	CX	число микки на 8 пикселов по горизонтали
	DX	число микки на 8 пикселов по вертикали
Возвращаемые значения:	нет	

Функция 16: Запретить появление курсора в специальной области

Функция 16 создает на экране условную специальную область. Если курсор попадает в эту область, то он исчезает. Для отмены действия специальной области необходимо произвести вызов функции 1.

Входные значения:	AX	0010h
	ES:DX	указатель на массив, определяющий специальную область
Возвращаемые значения:	нет	

Формат массива:

Смещение	Параметр
0000h	левая горизонтальная координата
0002h	верхняя вертикальная координата
0004h	правая горизонтальная координата
0006h	нижняя вертикальная координата

Функция 17: Задать параметры большого блока графического курсора

Примечание:

Функция определена для драйвера PC Mouse. Фирма Microsoft не дает документации по этой функции.

Действие функции 17 аналогично действию функции 9. Отличие состоит в том, что для функции 17 задается размер массивов, определяющих маски экрана и курсора.

Регистры 80i86

Входные значения:	AX	0011h
	BH	ширина курсора в словах (2 байта)
	CH	количество рядов в курсоре по вертикали
	BL	горизонтальная координата горячего пятна курсора в маске курсора (от -16 до 16)
	CL	вертикальная координата горячего пятна курсора в маске курсора (от -16 до 16)
	ES:DX	указатель на массивы масок
		(BH*CH) слов — маска экрана
		(BH*CH) слов — маска курсора
Возвращаемые значения:	AH	0FFFFh (-1) функция выполнена успешно

Функция 18:
Не используется

Функция 19: Установить порог удвоенной скорости

Функция 19 устанавливает значение порога скорости движения "мыши" (измеряемой в микки в секунду), при превышении которого скорость движения кур-

сора на экране удваивается. По умолчанию величина порога удвоенной скорости составляет 64 микки в секунду.

Регистры 80i86

Входные значения:	AX	0013h
	DX	порог скорости в микки в секунду
Возвращаемые значения:	нет	

Функция 20: Установить временную подпрограмму обработки прерывания

Функция 20 позволяет вам временно установить новую подпрограмму обработки прерываний, поступающих от "мыши", либо просто изменить маску вызова (см. ф.12). Адрес входа в старую подпрограмму должен быть обязательно восстановлен перед тем, как ваша прикладная программа закончит свою работу.

Регистры 80i86

Входные значения:	AX	0014h
	BX:DX	FAR указатель на новую подпрограмму
	CX	новая маска вызова (см. ф.12) бит 0 изменение позиции курсора бит 1 нажата левая клавиша бит 2 отпущена левая клавиша бит 3 нажата правая клавиша бит 4 отпущена правая клавиша бит 5 нажата средняя клавиша (Mouse Systems) бит 6 отпущена средняя клавиша (Mouse Systems) биты 7-15 не используются
Возвращаемые значения:	BX:DX	FAR адрес старой подпрограммы обработки прерываний
	CX	маска вызова старой подпрограммы

Эта функция в своей работе использует регистры AX, BX, CX, DX, SI, DI, DS и ES.

Следующие три функции (21, 22 и 23) служат для сохранения и последующего восстановления параметров состояния драйвера "мыши". Это необходимо в случае запуска изнутри вашей прикладной программы другой программы, тоже использующей "мышь".

Функция 21: Получить данные о размере буфера для записи состояния драйвера

Функция 21 возвращает данные о размере необходимого буфера, в который будут записаны параметры текущего состояния драйвера "мыши", с целью их последующего восстановления.

Регистры 80i86

Входные значения:	AX	0015h
Возвращаемые значения:	BX	размер буфера, необходимого для записи текущего состояния драйвера (байт)

Функция 22: Записать параметры текущего состояния драйвера в буфер

Функция 22 записывает параметры текущего состояния драйвера "мыши" в буфер, с целью их последующего восстановления. Размер необходимого буфера определяется при помощи функции 21.

Регистры 80i86

Входные значения:	AX	0016h
	ES:DX	указатель на буфер
Возвращаемые значения:	нет	

Функция 23: Восстановить параметры состояния драйвера из буфера

Функция 23 позволяет восстановить параметры предыдущего состояния драйвера "мыши", записанные ранее в буфер (см. фф. 21, 22).

Регистры 80i86

Входные значения:	AX	0017h
	ES:DX	указатель на буфер
Возвращаемые значения:	нет	

Функция 24: Установить альтернативную подпрограмму обработки прерываний

Функция 24 работает аналогично функции 12. Она позволяет установить альтернативную подпрограмму обработки тех прерываний "мыши", которые не были включены в подпрограмму, установленную функцией 12. Вы можете установить до трех различных подпрограмм обработки прерываний, путем последовательных обращений к функции 24.

Регистры 80i86

Входные значения:	AX	0018h
	CX	маска вызова бит 0 изменение позиции курсора бит 1 нажата левая клавиша бит 2 отпущена левая клавиша бит 3 нажата правая клавиша бит 4 отпущена правая клавиша бит 5 клавиша Shift была нажата в момент прерывания бит 6 клавиша Ctrl бы-

		ла нажата в момент прерывания бит 7 клавиша Alt была нажата в момент прерывания биты 8-15 не используются
	ES:DX	FAR указатель на подпрограмму обработки прерывания
Возвращаемые значения:	AX	0FFFFh - ошибка

Установленный бит (1) разрешает прерывание, сброшенный (0) — запрещает.

Во время вызова подпрограммы обработки прерывания драйвер передает ей следующие параметры:

AH	маска условия (биты установлены так же, как в маске вызова)
BX	состояние клавиш
CX	горизонтальная координата курсора
DX	вертикальная координата курсора
DI	значение вертикального счетчика микки
SI	значение горизонтального счетчика микки

Функция 25: Получить вектор прерывания, указывающий на установленную пользователем подпрограмму обработки

Функция 25 осуществляет попытку найти установленную пользователем подпрограмму обработки прерывания (определенную функцией 24), маска вызова которого соответствует маске, содержащейся в CX.

Регистры 80i86

Входные значения:	AX	0019h
	CX	маска вызова (аналогично функции 24)
Возвращаемые значения:	AX	0FFFFh вектор или маска не найдены
	BX:DX	указатель на вектор прерывания пользователя (0 если AX=0FFFFh)
	CX	маска вызова (0 если AX=0FFFFh)

Функция 26: Установить порог чувствительности "мыши"

Функция 26 позволяет установить порог чувствительности "мыши" по горизонтальной скорости движения, вертикальной скорости движения, а также порог удвоенной скорости "мыши".

Регистры 80i86

Входные значения:	AX	001Ah
	BX	горизонтальная скорость
	CX	вертикальная скорость
	DX	порог удвоенной скорости в микки в секунду 0000h устанавливает стандартный порог 64/сек
Возвращаемые значения:		нет

Функция 27: Получить данные о чувствительности "мыши"

Функция 27 возвращает параметры, установленные функцией 26.

Регистры 80i86

Входные значения:	AX	001Bh
Возвращаемые значения:	BX	горизонтальная скорость
	CX	вертикальная скорость
	DX	порог удвоенной скорости в микки в секунду

Функция 28: Установить частоту прерываний "мыши"

Функция 28 позволяет установить частоту прерываний, вырабатываемых аппаратными средствами "мыши" для передачи драйверу информации о своем состоянии и накопленном числе сигналов микки.

Регистры 80i86

Входные значения:	AX	001Ch
	BX	желаемая частота прерываний (BL) 00h прерывания запрещены 01h 30 прерываний в секунду 02h 50 прерываний в секунду 03h 100 прерываний в секунду 04h 200 прерываний в секунду 05h-FFh не определено
Возвращаемые значения:		нет

Если функции передается значение большее 04h (BL), драйвер аппаратной поддержки Microsoft InPort может вести себя непредсказуемо.

Функция 29: Установить страницу дисплея

Функция 29 переключает экран на указанную страницу (курсор при этом становится видимым).

Регистры 80i86

Входные значения:	AX	001Dh
	BX	номер страницы дисплея (0-7)
Возвращаемые значения:		нет

Функция 30: Получить номер текущей страницы дисплея

Регистры 80i86

Входные значения:	AX	001Eh
Возвращаемые значения:	BX	номер текущей страницы дисплея

Функция 31: Запретить работу драйвера "мыши"

Функция 31 запрещает работу драйвера "мыши", восстанавливая значения векторов прерываний 33h, 10h и 71h (для процессора 8086) или 74h (для процессоров 80286/386). Функция возвращает значение старого вектора прерывания 33h, указывающего либо на интерфейс "мыши" более высокого уровня (BX:CX), либо первоначальное значение этого вектора (ES:BX). В последнем случае работа драйвера "мыши" будет полностью запрещена.

Регистры 80i86

Входные значения:	AX	001Fh
Возвращаемые значения:	AX	001Fh функция выполнена успешно 0FFFFh ошибка
	BX:CX	старый вектор прерывания 33h, указывающий на интерфейс "мыши" более высокого уровня
	ES:BX	первоначальное значение вектора прерывания 33h

Функция 32: Восстановить работу драйвера "мыши"

Функция 32 восстанавливает прежние значения векторов прерываний 10h и 71h (8086) или 74h (80286/386), измененные функцией 31.

Регистры 80i86

Входные значения:	AX	0020h
Возвращаемые значения:	нет	

Функция 33: Произвести начальную установку программного обеспечения "мыши"

Действие функции 33 аналогично действию функции 0 с той разницей, что функция 33 не производит начальной установки аппаратных средств "мыши".

Регистры 80i86

Входные значения:	AX	0021h
Возвращаемые значения:	AX	0021h драйвер "мыши" не установлен 0FFFFh драйвер "мыши" установлен
	BX	0002h начальная установка произведена

Функция 34: Выбрать язык для выдачи диагностических сообщений

Регистры 80i86

Входные значения:	AX	0022h
	BX	код языка (BL) 00h Английский 01h Французский 02h Голландский 03h Немецкий 04h Шведский 05h Финский 06h Испанский 07h Португальский 08h Итальянский другие значения не используются
Возвращаемые значения:	нет	

Значения, отличные от 00h, могут быть использованы только для драйвера Microsoft International Mouse.

Функция 35: Получить информацию об используемом языке

Регистры 80i86

Входные значения:	AX	0023h
Возвращаемые значения:	BX	код языка (BL)
См. функцию 34		

Функция 36: Получить дополнительную информацию о "мыши"

Регистры 80i86

Входные значения:	AX	0024h
Возвращаемые значения:	AX	0FFFFh ошибка, иначе BH:BL номер версии драйвера
	CH	тип интерфейса "мыши" 01h "мышь", подключаемая к общей шине 02h "мышь", подключаемая к последовательному порту 03h "мышь", подключаемая к Microsoft InPort 04h "мышь", подключаемая к IBM PS/2 Pointing Device Port 05h "мышь" Hewlett-Packard
	CL	номер запроса прерывания (IRQ) 00h PS/2 позиционирующее устройство 01h не определено 02h IRQ2 03h IRQ3 ... ... 07h IRQ7

Библиотека регистрового интерфейса EGA (RIL)

Следующие 9 функций добавлены к прерыванию 10h BIOS (базовой системы ввода-вывода) для поддержки работы библиотеки регистрового интерфейса EGA. Эти функции позволяют вашей программе производить запись и чтение из регистров видеоадаптера EGA, первоначально предназначенных только для записи.

240	0F0h	Прочитать данные из отдельного регистра
241	0F1h	Записать данные в отдельный регистр
242	0F2h	Прочитать данные из последовательной группы регистров
243	0F3h	Записать данные в последовательную группу регистров
244	0F4h	Прочитать данные из группы отдельных регистров
245	0F5h	Записать данные в группу отдельных регистров
246	0F6h	Установить все регистры в их начальные значения
247	0F7h	Задать начальные значения для регистров
248, 249	не используются	
250	0FAh	Получить информацию о состоянии RIL

В качестве входных данных для функций RIL используются следующие номера и адреса портов ввода-вывода и регистров EGA:

Номер порта	Название	Количество регистров	Индексы регистров	Регистр адреса
00h	Контроллер электронно-лучевой трубки	25	0-24	3x4h
08h	Распределитель (Sequencer)	5	0-4	3C4h
10h	Графический контроллер	9	0-8	3CEh
18h	Контроллер атрибутов	20	0-19	3C0h
Одиночные регистры				Адрес
20h	Комплексный регистр вывода	1	не исп.	3C2h
28h	Регистр управления признаками	1	не исп.	3xAh
30h	Регистр 1 графической позиции	1	не исп.	3CCh
38h	Регистр 2 графической позиции	1	не исп.	3CAh

Примечание: x=B или D в зависимости от базового адреса ввода-вывода, определяемого битом 0 Комплексного регистра вывода (B для монохромного режима, D для цветного режима).

<u>Функция 240:</u> Прочитать данные из отдельного регистра EGA

Регистры 80i86

Входные значения:	AH	0F0h
	BX	индекс регистра
	DX	номер порта
Возвращаемые значения:	BL	данные
		значения остальных регистров восстанавливаются

<u>Функция 241:</u> Записать данные в отдельный регистр EGA

Регистры 80i86

Входные значения:	AH	0F1h
	BL	индекс регистра (для неодиночных регистров) данные (для одиночных регистров)
	BH	данные (для неодиночных регистров) игнорируется (для одиночных регистров)
	DX	номер порта
Возвращаемые значения:		регистры BH и DX не восстанавливаются, значения остальных регистров восстанавливаются

<u>Функция 242:</u> Прочитать данные из последовательной группы регистров указанного порта EGA

Регистры 80i86

Входные значения:	AH	0F2h
	CH	номер стартового индекса
	CL	количество регистров (должно быть > 1)
	DX	номер порта
	ES:BX	указатель на буфер-приемник (CL байт)
Возвращаемые значения:		данные в буфере значение CX не восстанавливается, значения остальных регистров восстанавливаются

<u>Функция 243:</u> Записать данные в последовательную группу регистров указанного порта EGA

Регистры 80i86

Входные значения:	AH	0F3h
	CH	номер стартового индекса
	CL	количество регистров (должно быть > 1)
	DX	номер порта
	ES:BX	указатель на буфер-источник данных (CL байт)
Возвращаемые значения:		значения BX, CX, DX не восстанавливаются, значения остальных регистров восстанавливаются.

<u>Функция 244:</u> Прочитать данные из группы отдельных регистров.

Регистры 80i86

Входные значения:	AH	0F4h
	CX	количество регистров (должно быть > 1)
	ES:BX	указатель на таблицу описания (CL*4 байт) каждая запись в табли-

		це состоит из 4 байт: байты 0-1 номер порта байт2 номер индекса регистра байт 3 заполняется прочитанными данными
Возвращаемые значения:	данные в байте 3 соответствующей записи таблицы описания; значение CX не восстанавливается, значения остальных регистров восстанавливаются	

<u>Функция 245:</u> Записать данные в группы отдельных регистров

Регистры 80i86

Входные значения:	AH	0F5h
	CX	количество регистров (должно быть > 1)
	ES:BX	указатель на таблицу описания (CL*4 байт) каждая запись в таблице состоит из 4 байт: байты 0-1 номер порта байт 2 номер индекса регистра байт 3 данные для вывода в регистр
Возвращаемые значения:	значение CX не восстанавливается, значения остальных регистров восстанавливаются	

<u>Функция 246:</u> Установить все регистры в их начальные значения

Функция 246 устанавливает все регистры EGA в их начальные значения, определяемые функцией 247.

Регистры 80i86

Входные значения:	AH	0F6h
Возвращаемые значения:	значения всех регистров восстанавливаются	

<u>Функция 247:</u> Задать начальные значения для регистров

Функция 247 передает драйверу начальные значения регистров EGA, которые в дальнейшем будут использоваться функцией 246 по умолчанию.

Регистры 80i86

Входные значения:	AH	0F7h
	CX	флаг выбора цвета VGA 5448h позволяет EGA RIL считать байт со смещением 14h в таблице, на которую указывает ES:BX, как значение для Регистра выбора цвета Контроллера атрибутов VGA
	DX	номер порта
	ES:BX	таблица начальных значений регистров каждый байт таблицы последовательно задает значение соответствующему регистру порта (значения всех регистров порта должны быть определены)
Возвращаемые значения:	значения BX и DX не восстанавливаются, значения остальных регистров восстанавливаются	

<u>Функции 248, 249 не используются.</u>

<u>Функция 250:</u> Получить информацию о состоянии RIL

Функция 250 позволяет пользователю определить, установлена библиотека RIL EGA или нет. При вызове функции значение BX должно быть равно нулю. Если по возвращении из функции значение BX осталось равным нулю, то RIL не установлена.

Регистры 80i86

Входные значения:	AH	0FAh
	BX	0000h
Возвращаемые значения:	AX	восстанавливается
	BX	0000h если RIL не установлена
	ES:BX	указатель на номер версии программного обеспечения RIL EGA

Ниже приводится текст программы "Пианино", в которой используются функции программирования "мыши".

```
/* Turbo C.
 * Программа "Пианино", авторы В.Боковой, А.Синев.
 * Программа выводит на экран клавиатуру пианино, из которого
 * можно извлекать звуки при помощи мыши.
 * Расчет частоты тона производится по формулам:
 *      частота_До = 32.75 * (2^(номер_октавы-1)),
 * для нот текущей октавы
 *      частота_ноты_i = частота_До * (2^(i/12)),
 * где номер_октавы = 0 соответствует субконтроктаве.
 * Программа рассчитана на работу с видеоадаптерами EGA или VGA.
 */

#include <stdio.h>
#include <stdlib.h>
#include <graphics.h>
#include <dos.h>
#include <conio.h>
#include <math.h>

#define Radix 10                    /* Основание системы счисления */

void play(void);                    /* Работа с мышью */
void oct_freq(int);                 /* Расчет частот нот октавы */
```

```
void show_freq(int);                /* Вывести частоту тона на экран */
void hide_cursor(void);             /* Сделать курсор невидимым */
void init_graph(void);              /* Инициализация графической */
                                    /* системы */
int init_mouse(void);               /* Инициализация мыши */
void set_cursor_position(void);     /* Установить курсор в заданную */
                                    /* позицию */
void set_mickey_pixel(void);        /* Установить соотношение микки-*/
                                    /* пиксел */
void set_cursor_shape(void);        /* Задать параметры курсора */
void show_cursor(void);             /* Сделать курсор видимым */
void read_cursor(void);             /* Определить местоположение */
                                    /* курсора */

unsigned int notes[12];             /* Массив частот нот текущей */
                                    /* октавы */

struct REGPACK ioregs;              /* Регистры 80i86 */
/*******************************************************************/

void main(void)
{
  register int i;                   /* Счетчик */

  init_graph();                     /* Инициализация графической */
                                    /* системы */
  init_mouse();                     /* Инициализация мыши */
  set_cursor_shape();               /* Задать параметры курсора */
  set_cursor_position();            /* Установить курсор в заданную */
                                    /* позицию */
  set_mickey_pixel();               /* Установить соотношение микки-*/
                                    /* пиксел */
  setfillstyle(SOLID_FILL, BLUE);/* Задать модель вывода */
  bar(0, 0, 639, 349);              /* Заполнить экран синим */
  setcolor(WHITE);                  /* Установить белый цвет вывода */
  rectangle(0, 0, 639, 349);        /* Обвести рамку */
  setfillstyle(SOLID_FILL, YELLOW);          /* Задать модель вывода */
  bar(0, 100, 639, 200);            /* Нарисовать панель клавиатуры */
  setcolor(RED);                    /* Установить красный цвет */
  for (i = 2; i < 50; i++)          /* Разметить "белые" клавиши */
    line(11+12*i, 100, 11+12*i, 200);
  setfillstyle(SOLID_FILL, BLACK);          /* Задать черный цвет */
  for (i = 0; i < 7; i++) {         /* Разметить черные клавиши */
    if (i)
      bar( 7+84*i, 100, 15+84*i, 150);
    bar(31+84*i, 100, 39+84*i, 150);
    bar(43+84*i, 100, 51+84*i, 150);
    bar(67+84*i, 100, 75+84*i, 150);
    bar(79+84*i, 100, 87+84*i, 150);
    bar(91+84*i, 100, 99+84*i, 150);
  }
  setcolor(GREEN);                  /* Установить зеленый цвет */
  for (i = 0; i <= 7; i++)          /* Разметить октавы */
    line(23+84*i, 100, 23+84*i, 200);
  play();                           /* Функция работы с мышью */
  closegraph();                     /* Закрыть графическую систему */
} /* main */

/*******************************************************************/

void play(void)                     /* Работа с мышью */
{
  int oct=0, oldoct=0;              /* Номер октавы, старая октава */
  unsigned note;                    /* Частота текущей ноты */
  char buffer[5];                   /* Буфер для выводимого текста */

  setfillstyle(SOLID_FILL, RED);  /* Задать модель вывода */
  bar(285, 40, 345, 70);            /* Нарисовать прямоугольник */
  setcolor(WHITE);                  /* Установить белый цвет */
settextstyle(TRIPLEX_FONT, HORIZ_DIR, 3);  /* Характеристики */
                                    /* текста */
outtextxy(290, 42, "QUIT");         /* Вывести текст */
show_cursor();                      /* Сделать курсор видимым */
while ( 1 ) {                       /* Бесконечный цикл */
  ioregs.r_bx = 0;                  /* Инициализация */
  while (ioregs.r_bx == 0) {        /* Пока клавиша не нажата */
    read_cursor();                  /* Прочитать состояние курсора */
                                    /* и мыши */
    /* Вывести координаты курсора на экран */
    printf("%3d %3d\r", ioregs.r_cx, ioregs.r_dx);
    /* Ограничения в области нижних и верхних частот */
    if (ioregs.r_cx > 24 && ioregs.r_cx < 613)
          oct = (ioregs.r_cx - 24) / 84 + 1;  /* Рассчитать номер */
                                              /* октавы */
    if ( oldoct != oct ) {
          oct_freq(oct);            /* Рассчитать частоты нот октавы */
          itoa(notes[0], buffer, Radix);  /* Преобразовать целое в */
                                          /* строку */
          setfillstyle(SOLID_FILL, RED);  /* Установить цвет */
          settextstyle(DEFAULT_FONT, HORIZ_DIR, 1);
                    /*Характеристики текста */
          hide_cursor();            /* Сделать курсор невидимым */
          bar(295, 80, 335, 95); /* Прямоугольник */
          bar(295, 6, 316, 20);  /* Прямоугольник */
          setcolor(WHITE);          /* Белый цвет */
          outtextxy(300, 85, buffer);   /* Вывести "До" текущей */
                                        /* октавы */
          itoa(oct, buffer, Radix);     /* Преобразовать номер */
                                        /* октавы в строку */
          outtextxy(302, 10, buffer);   /* Вывести номер октавы */
          show_cursor();            /* Сделать курсор видимым */
          oldoct = oct;             /* Сохранить номер октавы */
    }
  }
  if ( ioregs.r_bx == 1 ) {         /* Если нажата левая клавиша */

    /* Если курсор находится в области клавиатуры */
    if (ioregs.r_dx > 100 && ioregs.r_dx < 200) {

          /* Ограничения в области нижних и верхних частот */
          if (ioregs.r_cx > 24 && ioregs.r_cx < 613) {

            /* Привести горизонтальные координаты к эталонной */
            /* октаве */
            ioregs.r_cx = ioregs.r_cx - 24 - ( oct - 1 ) * 84;

            /* Определение местоположения курсора и частоты */
            /* текущего тона */

            /* Для белых клавиш */
            if (ioregs.r_cx > 70 && ioregs.r_cx < 83)
              note = notes[11];
            else {
              if (ioregs.r_cx > 59 && ioregs.r_cx < 70)
                note = notes[9];
              else {
                if (ioregs.r_cx > 47 && ioregs.r_cx < 59)
                      note = notes[7];
                else {
                      if (ioregs.r_cx > 35 && ioregs.r_cx < 47)
                        note = notes[5];
                      else {
                        if (ioregs.r_cx > 23 && ioregs.r_cx < 35)
                          note = notes[4];
                        else {
                          if (ioregs.r_cx > 11 && ioregs.r_cx < 23)
                            note = notes[2];
                          else
```

```
                                if (ioregs.r_cx > 0 && ioregs.r_cx < 11)
                                note = notes[0];
                        }
                    }
            }
        }
    }
    if (ioregs.r_dx < 150) {            /* Для черных клавиш */
      if (ioregs.r_cx >= 67 && ioregs.r_cx <= 75)
        note = notes[10];
      else {
        if (ioregs.r_cx >= 55 && ioregs.r_cx <= 63)
              note = notes[8];
        else {
              if (ioregs.r_cx >= 43 && ioregs.r_cx <= 51)
                note = notes[6];
              else {
                if (ioregs.r_cx >= 19 && ioregs.r_cx <= 27)
                  note = notes[3];
                else
                  if (ioregs.r_cx >=  8 && ioregs.r_cx <= 14)
                    note = notes[1];
              }
        }
      }
    }
    show_freq(note);        /* Вывести частоту тона на экран */
    sound(note);            /* Включить звук */
    delay(200);             /* Временная задержка */
    nosound();              /* Выключить звук */
  }
 }
}
 if ( ioregs.r_bx == 2 ) {          /* Нажата правая клавиша */
   if (ioregs.r_cx>285 && ioregs.r_dx>40 &&
                ioregs.r_cx<345 && ioregs.r_dx<70)
       return;      /* Возврат, если курсор в области "Quit" */
 }
}
} /* play */

/**********************************************************************
/

void oct_freq(int oct)              /* Расчет частот нот октавы */
{
  double f[12];                     /* Частоты нот октавы oct */
  register int i;                   /* Счетчик */

  f[0] = 32.75 * pow(2.0, oct-1);  /* Частота "До" */
  notes[0] = (unsigned)f[0];        /* Целая часть частоты */

  for (i = 1; i <= 11; i++) {       /* Частоты остальных нот октавы */
    f[i] = f[0] * pow(2.0, i/12.0);
    notes[i] = (unsigned)f[i];      /* Целая часть частоты */
  }
} /* oct_freq */

/**********************************************************************
/

void show_freq(int fr)              /* Вывести частоту тона на экран */
{
  char buffer[5];                   /* Буфер для текста */

  settextstyle(DEFAULT_FONT, HORIZ_DIR, 1);  /* Установить */
                                    /* характеристики */
                                    /* выводимого текста */
  itoa(fr, buffer, Radix);                /* Преобразовать в строку */
  setfillstyle(SOLID_FILL, RED);          /* Задать модель вывода */
  hide_cursor();                          /* Сделать курсор невидимым */
  bar(295, 205, 335, 220);                /* Нарисовать прямоугольник */
  setcolor(WHITE);                        /* Цвет текста */
  outtextxy(300, 210, buffer);            /* Вывести строку на экран */
  show_cursor();                          /* Сделать курсор видимым */
} /* show_freq */

/**********************************************************************
/

void hide_cursor(void)                    /* Сделать курсор невидимым */
{
  ioregs.r_ax = 2;                        /* Функция 2 "Сделать курсор */
                                          /* невидимым */
  intr(0x33, &ioregs);                    /* Прерывание 33h */
} /* hide_cursor */

/**********************************************************************
**/

void init_graph(void)                     /* Инициализация графической */
                                          /* системы */
{
  /* Графический драйвер, графический режим, код ошибки */
  int gdriver = EGA, gmode = EGAHI, errorcode;

  initgraph(&gdriver, &gmode, "D:\\TC\\BGI");           /*
Инициализация системы */
  errorcode = graphresult();              /* Результат инициализации */
  if( errorcode != grOk ) {               /* Ошибка при инициализации */
    printf(" Graphics System Error: %s\n", grapherrormsg(errorcode));
    exit(1);                              /* Выход из программы по ошибке */
  }
} /* init_graph */

/*********************************************************************/

int init_mouse(void)                      /* Инициализация мыши */
{
  ioregs.r_ax = 0;                        /* Функция 0 */
                                          /* "Начальная установка */
                                          /* драйвера мыши" */
  intr(0x33, &ioregs);                    /* Прерывание 33h */
  if ( ioregs.r_ax == 0 ) {               /* Если AX = 0, то выход */
    printf("MOUSE NOT INSTALLED ! PRESS ANY KEY !\n");
    getch();                              /* Подождать нажатия клавиши */
    closegraph();                         /* Закрыть графическую систему */
    exit(1);                              /* Закончить работу программы */
  }
  return 0;                               /* Возврат, если драйвер */
                                          /* установлен */
} /* init_mouse */

/**********************************************************************
/

void set_cursor_position(void)            /* Установить курсор в заданную */
                                          /* позицию */
{
  ioregs.r_ax = 4;                        /* Функция 4 "Установить курсор */
                                          /* в заданную позицию на экране" */
  ioregs.r_cx = 315;                      /* Горизонтальная координата */
                                          /* курсора */
  ioregs.r_dx = 160;                      /* Вертикальная координата */
                                          /* курсора */
  intr(0x33, &ioregs);                    /* Прерывание 33h */
} /* set_cursor_position */
```

```
/******************************************************************
/

void set_mickey_pixel(void)       /* Установить соотношение */
                                  /* микки-пиксел */
{
  ioregs.r_ax = 15;               /* Функция 15 "Установить */
                                  /* соотношение микки-пиксел" */
  ioregs.r_cx = 36;               /* 36/8 по горизонтали */
  ioregs.r_dx = 36;               /* 36/8 по вертикали */
  intr(0x33, &ioregs);            /* Прерывание 33h */
} /* set_mickey_pixel */

/******************************************************************
/

void set_cursor_shape(void)       /* Задать параметры */
                                  /* курсора */
{
  unsigned int cursor[2][16];
                      /* Массив масок экрана */
                                  /* и курсора */
  unsigned far *fpuns;            /* FAR указатель на */
                                  /* массив */
  unsigned far **pfpuns;                  /* Указатель на */
                                  /* предыдущее */


/* Содержимое масок курсора и экрана */
  cursor[1][0]  = 0x0001;   /*  0000000000000001  */
  cursor[1][1]  = 0x0002;   /*  0000000000000010  */
  cursor[1][2]  = 0x000e;   /*  0000000000001110  */
  cursor[1][3]  = 0x003c;   /*  0000000000111100  */
  cursor[1][4]  = 0x00fc;   /*  0000000011111100  */
  cursor[1][5]  = 0x03f8;   /*  0000001111111000  */
  cursor[1][6]  = 0x0ff8;   /*  0000111111111000  */
  cursor[1][7]  = 0x3ff0;   /*  0011111111110000  */
  cursor[1][8]  = 0xfff0;   /*  1111111111110000  */
  cursor[1][9]  = 0x3fe0;   /*  0011111111100000  */
  cursor[1][10] = 0x1fe0;   /*  0001111111100000  */
  cursor[1][11] = 0x3fc0;   /*  0011111111000000  */
  cursor[1][12] = 0x78c0;   /*  0111100011000000  */
  cursor[1][13] = 0xf000;   /*  1111000000000000  */
  cursor[1][14] = 0xe000;   /*  1110000000000000  */
  cursor[1][15] = 0xc000;   /*  1100000000000000  */

  cursor[0][0]  = 0xfffc;   /*  1111111111111100  */
  cursor[0][1]  = 0xfff8;   /*  1111111111111000  */
  cursor[0][2]  = 0xffe0;   /*  1111111111100000  */
  cursor[0][3]  = 0xff81;   /*  1111111110000001  */
  cursor[0][4]  = 0xfe01;   /*  1111111000000001  */
  cursor[0][5]  = 0xf803;   /*  1111100000000011  */
  cursor[0][6]  = 0xe003;   /*  1110000000000011  */
  cursor[0][7]  = 0x8007;   /*  1000000000000111  */
  cursor[0][8]  = 0x0007;   /*  0000000000000111  */
  cursor[0][9]  = 0x800f;   /*  1000000000001111  */
  cursor[0][10] = 0xc00f;   /*  1100000000001111  */
  cursor[0][11] = 0x801f;   /*  1000000000011111  */
  cursor[0][12] = 0x021f;   /*  0000001000011111  */
  cursor[0][13] = 0x07ff;   /*  0000011111111111  */
  cursor[0][14] = 0x0fff;   /*  0000111111111111  */
  cursor[0][15] = 0x1fff;   /*  0001111111111111  */

  ioregs.r_ax = 9;                  /* Функция 9 "Задать параметры */
                                    /* графического курсора" */
  ioregs.r_bx = 16;                 /* Горизонтальная координата */
                                    /* горячего пятна */
  ioregs.r_cx = 0;                  /* Вертикальная координата */
                                    /* горячего пятна */

  fpuns = (unsigned far *)cursor;   /* Вычислить FAR адрес массива */
  pfpuns = &fpuns;                  /* Указатель на FAR адрес */
                                    /* массива */

  ioregs.r_es = *((unsigned *)pfpuns + 1);   /* Адрес сегмента в ES */
  ioregs.r_dx = *(unsigned *)pfpuns;         /* Смещение в DX */

  intr(0x33, &ioregs);              /* Прерывание 33h */
} /* set_cursor_shape */

/******************************************************************
/

void show_cursor(void)              /* Сделать курсор видимым */
{
  ioregs.r_ax = 1;                  /* Функция 1 "Сделать курсор
видимым" */
  intr(0x33, &ioregs);              /* Прерывание 33h */
} /* show_cursor */

/******************************************************************
/

void read_cursor(void)              /* Определить местоположение */
                                    /* курсора */
{                                   /* и состояние клавиш */
  ioregs.r_ax = 3;                  /* Функция 3 */
  intr(0x33, &ioregs);              /* Прерывание 33h */
}
```

А.Синев

По материалам:

W.Bright "Interface to Microsoft Mouse Driver", 1989.

"Programmer's Reference for Genius Mouse Driver", Genius Mouse Co., 1989.

"Using RS-232 Mouse for Personal Computer", RaisWare GmbH, 1989.

"VGA, Technical Reference", IBM, 1989.

D.Williams, R.Crisp "Mouse Programming", Personal Computer World, April 1990.

В предыдущем номере КомпьютерПресс был начат разговор о средствах графического отображения информации с описания пакетов инженерно-деловой графики. Сегодня продолжение темы — обзор пакетов иллюстративной графики, некоторых графических библиотек, а также универсальной системы HALO.

Графические пакеты

ПАКЕТЫ ИЛЛЮСТРАТИВНОЙ ГРАФИКИ

Пакет GraFIX Partner, версия 1.8
Фирма Brightbill - Roberts

Одним из самых простых пакетов иллюстративной графики является GraFIX Partner. Пакет GraFIX Partner предназначен для создания рисунков с использованием графического адаптера типа CGA. Пакет работает как с цветными, так и с монохроматическими дисплеями. К основным его достоинствам следует отнести то, что он занимает мало памяти на магнитном носителе (50 Кбайт), и требует мало оперативной памяти при своей работе. В основном пакет предназначен для быстрого создания несложных рисунков с разрешением 320x200 точек (CGA). Малый объем пакета и его низкая стоимость позволяют поставлять его с рисунками для иллюстрации и описания других пакетов, книг и т.д. Вся система состоит из одного файла Grf.exe. Старт осуществляется запуском этой программы. После запуска появляется полноэкранное меню, организованное по принципу заглавной буквы, — для выбора пункта меню нужно нажать клавишу, соответствующую выделенной букве в тексте пункта меню. Чтение рисунка с магнитного носителя и запись на него осуществляются посредством подменю, вызываемого клавишей I (I/O). Данное меню, кроме чтения и записи, дает возможность вывода рисунка на принтер в графическом режиме. В подменю печати, вызываемом нажатием клавиши "?", имеются пункты увеличения и уменьшения копии рисунка на бумаге, разворота на 180 градусов, осуществления черновой и чистовой печати, также предусмотрена возможность инвертирования цветов изображения при печати. Работа программы ориентирована на черно-белый принтер. Просмотр и редактирование рисунков осуществляются из главного меню путем нажатия клавиши "+" дополнительной клавиатуры, прекращение редактирования (просмотра) — нажатием клавиши "-". При создании рисунка к нужной точке подгоняется курсор и нажимается клавиша ENTER.При этом выбранная точка подсвечивается установленным ранее цветом. Повторное нажатие клавиши ENTER приводит к стиранию данной точки. В пакете предусмотрена возможность проведения отрезков и окружностей. Все эти действия выполняются через главное меню. Выход из пакета производится нажатием клавиши ESC. GraFIX Partner можно использовать как резедентную программу для запоминания текущего изображения на экране. Вызов резедентной части пакета поизводится по комбинации клавиш Ctrl + Alt + Plus.

Пакет Paintbrush IV, версия 4.0
Фирма Microsoft

Наиболее известным представителем описываемого класса является пакет Paintbrush , ранние версии которого очень часто включались в состав программного обеспечения, поставляемого вместе с персональным компьютером. Пакет предназначается для создания и оформления изображений на экране и вывода их на принтер. С его помощью пользователь может строить изображения путем компоновки их из ранее созданных, объединять изображения с текстами, изменять цвета и т.п. В пакете реализованы функции, позволяющие:

— манипулировать произвольными частями изображений;

— пользоваться библиотекой готовых изображений;

— наносить линии различного цвета;

— выводить текст разнообразными шрифтами;

— запоминать сформированные иллюстративные материалы на диске, осуществлять их поиск и воспроизведение.

Выполнение рисунков производится при помощи “мыши“. Пакет Paintbrush управляется с помощью двух иерархических меню. В первое меню, расположенное с левого края экрана, вынесены наиболее часто употребляемые операции. В меню эти операции представлены в виде пиктограмм. Для выбора операции нужно перевести курсор на соответствующую пиктограмму и нажать клавишу “мыши“. Во второе меню, расположенное вверху экрана, вынесены операции, которые реже встречаются в процессе создания изображения. Меню палитры цветов расположено внизу экрана. Данный пакет позволяет выводить созданное изображение на печатающее устройство в произвольном масштабе. Конфигурация пакета и установка типов периферийных устройств производится при генерации. Изменение типов периферийных устройств из программы не предусмотрено. Пакет поддерживает адаптеры от двуцветного CGA до шестнадцатицветного VGA. Для VGA поддерживается шестнадцатицветный формат 752x410 и двухцветный 1280x800.

Пакет 35mm Express, версия 4.1 (1988 год) Фирма Business & Professional Software.

Пакет иллюстративной и деловой графики 35mm Express работает с адаптерами типа CGA и EGA. Для каждого адаптера имеется свой набор файлов. Возможен вывод изображений на принтер, подключенный как через параллельный, так и через последовательный порты. Допускается работа с четырьмя типами матричных принтеров: IBM и EPSON (9- и 24-игольчатые). Вывод на графопостроитель не предусмотрен. Необходимо подчеркнуть такую особенность пакета как организация хранения информации с использованием фреймового подхода. В каждом фрейме хранится информация о нескольких изображениях, причем фреймы определяются типом информации (графики, диаграммы, рисунки, надписи и т.д.) и набором образцов для создания рисунков. Допускается создание фреймов только определенных размеров — 32, 80, 120 и 320 Кбайт, что не совсем удобно. Общение с пакетом осуществляется с помощью ниспадающих меню. Главное меню содержит *5* пунктов:

— создание и редактирование рисунков;

— печать рисунков;

— операции с фреймами;

— операции с образцами;

— дополнительные сервисные операции.

В режиме редактирования 35mm Express осуществляет выбор из фрейма необходимого набора рисунков и создание на его базе нового набора. Каждый набор содержит определенную конфигурацию рисунков. Подсветив имя рисунка с помощью курсора и нажав клавишу ENTER (или F10), пользователь попадает в собственно режим редактирования (создания) рисунка. Все надписи, заголовки, исходные данные вводятся в специальную таблицу спецификации рисунка. С помощью меню, которое появляется в правом верхнем углу экрана, пользователь имеет возможность осуществлять операции с отдельными объектами на рисунке и с группами объектов. Операции манипулирования включают в себя удаление, копирование и перемещение объектов. Перед проведением операций манипулирования требуется пометить объект с помощью курсора, а группу объектов — с помощью прямоугольника, который появляется на экране при вызове режима идентификации. Помимо вышеперечисленных возможностей, меню редактирования предоставляет возможность увеличения (уменьшения) объекта, передвижения его в вертикальном или горизонтальном направлении, изменения цвета, оттенка, тени для любой надписи или фигуры на рисунке. Как уже отмечалось ранее, цвет, величина букв и другие параметры берутся по умолчанию из того образца, в котором строится рисунок.

Для осуществления печати необходимо сначала выбрать соответствующий фрейм и в нем рисунок. После этого на экране появится меню печати, с помощью которого можно осуществить:

а) выбор принтера из четырех предоставляемых: IBM — 9 иголок, IBM — 24 иголки, EPSON — 9 иголок, EPSON — 24 иголки.

б) установку различных режимов для принтера, в том числе печать инвертированного изображения, прогонка листов бумаги и т.д.

в) установку каналов вывода (параллельный или последовательный).

Процесс трансляции изображения рисунка в управляющий код для принтера осуществляется как до начала печати, так и во время нее, что сокращает время создания копии рисунка на бумаге. Возможность подключения плоттера в пакете 35mm Express не предусмотрена.

Следующим подпунктом меню являются операции с фреймами. Они позволяют загружать в программу любые фреймы, просматривать таблицы содержимого фреймов, осуществлять удаление из фреймов различных рисунков и конфигураций, выполнять копирование рисунков из одного фрейма в другую, записывать фреймы на магнитные носители. В подпункте операций с образцами пользователь получает возможность редактирования существующих и создания новых образцов. Для создания нового образца нужно выбрать образец с именем NEW. Образцы содержат информацию о цветах слов, отдельных символов, линий, окружностей, оттенков, а также данные о размерах и типах градуировки и ширине используемых при рисовании линий. Имеется возможность создания надписей с указанием размера букв в различных направлениях и максимальной длины строк. Таким образом, в образце содержатся практически все необходимые установки для создания рисунка или графика. Через подпункт сервисных услуг пользователь осуществляет создание

новых фреймов, установку подключаемых шрифтов, смену обслуживаемых магнитных носителей, установку и просмотр текущих даты и времени. Пакет не поддерживает манипулятор "мышь".

В целом пакет 35mm Express предназначен для создания иллюстрированной документации (графиков, рисунков) при массовом тиражировании. Необходимо отметить великолепные цвета и разнообразные рисовальные возможности. Требуемая оперативная память — 256 Кбайт (320 Кбайт для DOS 3.0).

Пакет Ega Paint
Фирма RIX SoftWorks

Пакет иллюстративной графики Ega Paint предназначен для работы с графическим адаптером EGA. В нем реализованы основные функции, необходимые для создания и оформления изображения. Имеется возможность компоновки изображений из ранее созданных. Допускается работа с изображениями, созданными в других пакетах для адаптеров EGA, посредством использования резидентного модуля EGAPRINT.EXE, который может использоваться как в составе пакета, так и самостоятельно. Перед началом работы необходимо установить модуль EGAPRINT.EXE с указанием типа принтера и размера изображения. В течение работы данный модуль остается резидентным и его активизация производится одновременным нажатием клавиш Shift/Print Screen. Вывод изображений на графопостроитель невозможен. Поддерживается работа с "мышью". Управление осуществляется посредством меню, которое можно расположить в произвольном месте экрана.

Пакет FANTAVISION, версия 1.0 (1988 год)
Фирма Broderbund Software

Пакет FANTAVISION предназначен для создания мультфильмов. Он поддерживает четыре графических адаптера: EGA, CGA, Hercules и VGA. FANTAVISION позволяет создавать мультфильмы с количеством кадров, ограниченным только размерами оперативной памяти компьютера. Управление работой пакета осуществляется вложенными меню посредством "мыши". После запуска не экране возникает главное меню :
— ввод, вывод мультфильмов и их атрибутов;
— редактирование кадра мультфильма;
— манипуляция изображениями;
— выбор графического адаптера.

В режиме ввода-вывода пользователю предоставляется возможность загрузки готового мультфильма с внешнего носителя в оперативную память, а также задание фона кадров. Под фоном в данном случае понимается некоторое изображение, которое остается неизменным на протяжении всего мультфильма. В режиме редактирования кадров пользователь имеет возможность удалить, переместить, скопировать изображение, осуществить вставку фрагмента изображения в новое место, восстановить предыдущее состояние кадра мультфильма. В режиме манипулирования изображением осуществляются следующие действия:
— уменьшение и увеличение изображения вплоть до заполнения всего кадра;
— установка центра вращения кадров;
— вращение изображения как в его собственной плоскости вокруг оси Z, так и вокруг осей X и Y;
— деформация изображения вправо и влево.

Оси, вокруг которых осуществляется вращение, обязательно проходят через устанавливаемый пользователем центр вращения. Необходимо подчеркнуть возможность создания звукового сопровождения мультфильма. Это осуществляется посредством ввода амплитудных и фазовых характеристик звуковых сигналов, задаваемых графически с помощью "мыши". Создание мультфильмов осуществляется покадрово. Как правило, создание каждого кадра начинается с загрузки некоторого фонового изображения. При редактировании кадров возможно применение нескольких способов нанесения изображения: нанесение прямых линий и окружностей, закрашивание замкнутых контуров, выделение прямоугольного фрагмента изображения, ликвидация заданного фрагмента изображения, вывод текста на экран. В системе предусмотрена возможность создания и редактирования цветовой палитры. В состав пакета входит пограмма для прогона мультфильма отдельно от основной программы.

К недостаткам пакета FANTAVISION следует отнести необходимость загрузки фона для каждого нового кадра, отсутствие установки времени экспонирования кадра, ограниченность программы манипуляции кадром (возможна только манипуляция всем кадром), запоминание только одного предшествующего состояния. В пакете полностью отсутствуют режимы вывода изображения на такие внешние устройства как графопостроитель и принтер.

Пакет Show Partner
Фирма Bright-Roberts

Пакет предназначен для создания рисунков и мультипликации. Он состоит из четырех частей, организованных в виде отдельных загрузочных файлов:
— резидентной программы CAPTURE для считывания изображения с экрана;
— графического редактора GraFIX Editor;
— редактора сценария Script Editor;
— программы демонстрации SHOW.

Графический редактор GraFIX Editor позволяет быстро создавать и редактировать графические изображения с помощью "мыши", причем имеется возможность выхода из редактора, при котором редактор остается резидентным в памяти. Редактор сценария Script Editor, который используется для организации последовательности появления графических изображений на экране при демонстрации мультфильма. Он дает возможность использовать звуковые эффекты или изменять цвет изображения при демонстрации,

задавать время показа, изменять скорость демонстрации и способ появления изображения на экране. Программа SHOW необходим для демонстрации мультфильмов по готовым сценариям. Для этого достаточно указать имя файла, содержащего сценарий мультфильма. Для работы пакета необходим объем памяти на диске 1.2 Мбайт и объем оперативной памяти около 130 Кбайт.

Пакет IBM Storyboard Plus Фирма IBM

Пакет иллюстративной графики IBM Storyboard Plus позволяет помимо отдельных изображений создавать и мультфильмы. В состав пакета входят *5* функционально независимых модулей, каждый из которых может работать самостоятельно. Изображения в этом пакете можно создавать компоновкой уже имеющихся изображений и библиотечных примитивов при помощи графического редактора Picture Marker. В редакторе предусмотрена возможность увеличения части изображения с целью точной коррекции его отдельных деталей. Возможность создания собственной палитры путем смешивания в различной пропорции основных цветов позволяет значительно расширить цветовую гамму изображения. Для точного позиционирования отдельных частей изображения используется режим выдачи текущих координат курсора на поле экрана. Специальный редактор позволяет создавать шрифты, не входящие в комплект поставки. Редактор может управляться как с клавиатуры, так и при помощи "мыши".

Для построения сценария мультфильма применяется редактор Story Editor. Он позволяет указать последовательность появления изображений в мультфильме, скорость смены изображений, способ появления их на экране. При создании сценария можно рассматривать фрагменты мультфильма в непрерывном или пошаговом режиме. С помощью дополнительной аппаратуры возможно воспроизведение звуковых эффектов. Модуль Picture Maker позволяет использовать изображения, созданные другими программами, и вводить их с помощью телекамеры. Модуль Story Teller (ST.EXE) предназначен для демонстрации созданных мультфильмов, а текстовый редактор Text Marker (TM.EXE) служит для текстового оформления мультфильма.

Пакет поддерживает адаптеры CGA, EGA, MGA, разнообразные матричные и лазерные принтеры. Возможность выдачи на графопостроитель не предусмотрена. Пакет занимает на диске 2 Мбайта, а объем требуемой оперативной памяти составляет 256 Кбайт.

Таблица 1. Сравнительные характеристики пакетов иллюстративной графики.

Название пакета	Фирма изготовитель	Память на ЖМД	Тип адаптера	Наличие мультипликации	Поддержка манипулятора "мышь"	Удобство организации меню
Paintbrush	Microsoft	600 кБ	EGA	есть	есть	5
PC Illustrator	Compute Graphics Groop	380 кБ	CGA	есть	есть	4
35mm Express	Basines Professional Software	500 кБ	EGA,CGA	нет	нет	4
EGA Paint	RIX Softworks	500 кБ	EGA	нет	есть	3
Show Partner	Bright - Roberts & Co.	1,5 мБ	EGA	есть	есть	5
IBM Storyboad Plus	IBM	2 мБ	EGA	нет	есть	4

ПАКЕТЫ БИБЛИОТЕЧНОГО ТИПА

Наиболее типичными среди пакетов библиотечного типа являются WATFOR и TOOL KIT, более известный под названием EGATOOLS. Все эти пакеты предназначены для организации вывода графических изображений только на экран монитора. Следует отметить, что основная часть графических библиотек поставляется с соответствующими языками программирования.

Пакет Watfor
Фирма Watcom

Watfor представляет собой пакет иллюстративной графики и предназначен для создания графических изображений на экране монитора. Пакет состоит из графических библиотек, библиотек Фортран-77 и текстового редактора. Изображение строится на основе таких графических примитивов как линия, прямоугольник, круг, эллипс и т.п. Каждый из этих примитивов реализован программой графической библиотеки пакета. Для того, чтобы получить изображения этих фигур на экране, пользователь должен обратиться из своей программы, написанной на языке Фортран-77, к подпрограмме инициализации экрана, а затем к нужной ему подпрограмме из графической библиотеки. Задавая значения соответствующих параметров, можно перемещать и копировать изображение, изменять его размер и цвет. Имеется возможность задания и изменения палитры, а также наложения текстов на изображения. Пакет позволяет создавать качественные изображения и иллюстрации. Его использование возможно и при применении адаптера EGA. К недостаткам пакета можно отнести некоторую сложность его освоения. В пакете предусмотрена возможность создания исполняемого машинного кода, помещаемого в файл с расширением EXE. Для этого нужно указать ключ /EXE при компиляции.

Пакет EGATOOLS
Фирма Connell Scientific Graphics

EGATOOLS является высокоэффективным пакетом иллюстративной графики библиотечного типа и предназначается для создания изображений на экране. Пакет EGATOOLS — набор инструментальных средств, которые позволяют строить графическое изображение на основе геометрических примитивов, таких как прямая, прямоугольник, окружность, цветная замкнутая область. Эти примитивы характеризуются рядом параметров — координаты опорных точек, цвет, размеры, количество вершин и т.п. EGATOOLS позволяет строить и трансформировать элементарные геометрические фигуры, формировать изображение из отрезков прямой, перемещать графические объекты по экрану, копировать их, закрашивать и заштриховывать, накладывать одно графическое изображение на другое, налагать тексты на изображения, осуществлять демонстрацию созданных изображений. Пакет прост в освоении и использовании, что сделало его в СССР наиболее употребляемым среди систем библиотечного типа.

УНИВЕРСАЛЬНАЯ ГРАФИЧЕСКАЯ СИСТЕМА HALO

Пакет HALO (1988 год)
Фирма KYE International

В заключении рассмотрим пакет, который является универсальным в том смысле, что сочетает в себе функциональные возможности всех трех ранее рассмотренных классов.

Пакет HALO включает в себя программы для разносторонней обработки графической информации:

— графический редактор для создания рисунков произвольного вида;

— программу построения кусочно-линейных графиков, столбиковых и круговых диаграмм;

— программу создания и просмотра мультфильмов;

— библиотеку функций для использования их в программах на языках Си, Паскаль, Фортран в реализации фирмы Microsoft.

При этом рисунки, созданные с помощью программы построения графиков (диаграмм), могут быть использованы как основа либо как составные части при создании рисунка в графическом редакторе.

Пакет поддерживает рабуту с персональными компьютерами типа IBM PC XT/AT и PS/2. Возможна работа с 34 типами графических адаптеров, среди которых наиболее часто употребляемые: Hercules, CGA, EGA, VGA, SuperVGA. Для вывода информации на бумагу возможно использование одного из 26 типов принтеров и плоттеров. При выводе информации допукскается задание порта для подключения соответствующего устройства. Возможно также задать качество печати, режим вывода изображения в цвете, специальные режимы (центрирование изображения на бумаге, инвертирование цветов и т.п.). В качесве устройства для управления работой графического редактора можно использовать один из четырех манипуляторов. Чаще всего употребляется манипулятор типа "мышь". Перед работой с пакетом необходимо его настроить на соответствующее оборудование. Это осуществляется программой SETUP.EXE. После ее работы создается файл HALO.CNF, в котором хранятся произведенные установки.

Пакет HALO состоит из следующих составных частей:

1. HALO GRAPH (файл GS.EXE) — обеспечивает создание кусочно-линейных графиков, круговых и столбиковых диаграмм. Эта программа позволяет построить до 4 законченных комбинаций графиков на одном экране, при этом каждая комбинация может со-

держать до 12 различных графиков или диаграмм. Количество точек одного графика не может превышать 99. Программа предоставляет возможность создавать до 10 разлчных форм графиков и диаграмм. Вывод осей, градуировки и отцифровки их осуществляется автоматически (если они необходимы). Позиционирование камбинаций графиков на экране и установку их размера производится самим пользователем.

Управление работой HALO GRAPH осуществляется с помощью меню и функциональных клавиш. Выбор пункта меню производится нажатием клавиши, соответствующей выделенной букве нужного пункта. Главное меню программы позволяет создавать, редактировать, читать и записывать рисунок, осуществлять переформатирование данных любого рисунка, а также читать исходные данные из текстовых и двоичных файлов, производить запись рисунка в файл для использования его в графическом редакторе Dr. HALO III. При работе в любом меню данной программы имеется возможность просмотреть текущую комбиинацию графиков, либо все комбинации сразу (клавиши F2, F1), а также перейти к редактированию атрибутов активной в данный момент комбинации (установе осей, цветов, текстов, данных рисунка и т.д.).

Основным недостатком HALO GRAPH можно считать ограниченность числа точек для построения одного графика или диаграммы (99). Этот недостаток ограничивает применение программы в инженерных приложениях. Кроме этого фиксированность распорложения осей графиков и их градуировки также достаточно неудобна.

2. Dr. HALO III — предствляет собой графический редактор пакета HALO и используется для создания разнообразных рисунков. Запуск редактора осуществляется командой DRHALO. После запуска перед пользователем на экране появляются: в центре — рабочее поле рисунка, в левом верхнем углу — меню действий, в нижней части — меню цветов, штриховок для заполнения областей, а также меню толщины и форм линий создания любых объектов. Через меню действий пользователь получает возможность наносить надписи, проводить прямые и произвольные линии, осуществлять разворот выделенных в прямоугольниках объектов на 90 градусов, закрашивать замкнутые области и производить их штриховку, осуществлять несплошную закраску областей ("пульверизатор"), рисовать точечные объекты, уничтожать созданный объект. Кроме того в меню содержатся режимы чтения, записи и вывода рисунков на бумагу.

Управление работой Dr. HALO III производится установленным программой SETUP манипулятором (в частности, может быть использована "мышь"). Управление посредством клавиатуры невозможно. При работе с "мышью" крайняя левая кнопка служит для выбора действий из меню, смены цветов, толщин линий, их форм, а также для выполнения самих действий над рисунком, а правая кнопка служит для изменения размеров объекта действия (например, изменения размера окружности или прямоугольника, наносимых на рисунок). Размер необходимо устанавливать до создания объекта на экране. Удобным средсвом пакета является возможность отмены действия, произведенного на последнем шаге создания рисунка.

3. PRESENTS — данная часть HALO позволяет организовывать простые мультфильмы, кадры которых созданы графическим редактором или построителем графиков. При запуске программ появляется главное меню, позволяющее создать, запомнить, редактировать и показывать мультфильмы. При организации мультфильмов указывается время экспонирования кадра и координаты для его вывода. Недостаток этой программы состоит в том, что других атрибутов мультфильма таких как способ смены одного кадра другим, перемещение одного кадра по другому и т.д. не предусмотрено.

4. HALO´88 — представляет собой библиотеку функций, написанных на языках Си и Ассемблер и откомпиллированных трансляторами фирмы Microsoft. Библиотеки предназначены для использования в программах на языках совместимых с языком Microsoft C 4.0. В состав библиотек входят:

— функции установки графических режимов экрана в соответствии с набором адаптеров, поддерживаемых пакетом HALO (34 типа адаптеров);

— функции вывода в графическом режиме точек, линий, окружностей, секторов, прямоугольников и т.п.;

— функции установки цветов экрана, типов линий и штриховок;

— функции нанесения надписей на поле экрана различными наборами букв, входящими в состав пакета HALO (18 шрифтов);

— функции запоминания, чтения и вывода созданных изображений на бумагу для 26 типов периферийных устройств.

Наличие последней группы функций существенно выделяет HALO´88 среди других аналогичных библиотечных пакетов. Кроме того в программах, создаваемых пользователем нет необходимости дополнительных описаний функции.

Для сборки программ на этапе линковки необходимо указать имя библиотеки пакета в соответствии с используемой моделью памяти.

А.Смородинский, А.Воскресенский.

По материалам:

R.Jentz "Business Graphics Roundup", PC World, July 1988.
R.Goodwin "PC Paintbrush", PC World, November 1989.
P.Robinson " Buyer´s guide: business graphics", PC User, 8, November 21, 1989.

В этой статье приводится обзор специального выпуска сборника трудов Института инженеров электротехников и радиоэлектроников (США), посвященного разработке систем управления данными и знаниями (IEEE Transactions on knowledge and data engineering). Десять описанных в нем статей, безусловно, не отражают всего спектра научных исследований в области баз данных и баз знаний, особенно в столь кратком их изложении. Однако "КомпьютерПресс" считает своей обязанностью максимально восполнить нехватку информации о перспективных научных и исследовательских разработках в области вычислительной техники как за рубежом, так и в нашей стране и приглашает своих читателей к активному диалогу.

СУБД нового поколения

Концепция баз данных явилась закономерным отражением очередного этапа развития вычислительной техники, когда данные выделились в самостоятельный независимый от программ объект со своими законами движения и развития. Необходимость их изучения породила новое направление теории вычислительных систем — моделирование данных и его практическое приложение — системы управления базами данных (СУБД). Вопросами моделирования данных и создания СУБД занимались многие исследователи, однако, пожалуй, наиболее заметный след оставили в этой области работа Кодда, разработавшего реляционную модель данных, на основе которой сздано большинство современных коммерческих СУБД, и Чена, автора модели "сущность-связь", пожалуй, наиболее популярной объектно-ориентированной модели. Эти две модели, с одной стороны, ознаменовали собой построение законченной теории данных, несмотря на большое количество работ, появляющихся до сих пор в их развитие, с другой — показали ограниченность использования датологических моделей данных, в число которых входит и реляционная модель, при решении многих классов реальных задач. Проблема состоит в том, что кажущаяся универсальность моделей распространяется лишь на узкий класс задач управления организационными системами, т.е. приложений, которые характеризуются большими объемами относительно простых данных, и это не удивительно, поскольку моделирование данных развивалось одновременно с интенсивным внедрением вычислительной техники в сферу организационного управления.

В чем же проявилась ограниченность классического подхода к моделированию данных? Ответ на этот вопрос можно получить, если внимательно проанализировать особенности тех областей, в которые вычислительная техника стала вторгаться позже. Например, в настольных издательских системах требуется иметь аппарат для представления текста, графиков, пиктограмм и изображений в виде битовых карт. Системы автоматизированного проектирования и географические информационные системы, работающие с пространственными данными, нуждаются в представлении многоугольников, линий, точек и более сложных пространственных объектов. Приложения в области научных исследований часто используют в качестве данных временные ряды, матрицы, вектора.

Новые приложения могут потребовать изменения самих функций обработки данных, выполняемых СУБД. Например, в САПР инженер взаимодействует с СУБД, совершенствуя проект в течение недель или месяцев, чтобы затем сохранить его в качестве очередной модификации проекта и начать работу уже с этим

новым изделием. Такого рода транзакции не поддерживаются традиционными системами. Более того, работа с данными, описывающими пространственные объекты, требует, по сравнению с экономическими приложениями, использования значительно сложных ограничений целостности. Например, в процессе проектирования самолета должно накладываться такое ограничение целостности, которое запрещало бы размещать два агрегата в одном и том же месте.

Неуклонные изменения в технологии привели к снятию многих ограничений, на которые были рассчитаны традиционные коммерческие СУБД. Например, в связи со снижением стоимости запоминающих устройств с произвольной выборкой базы данных, которые ранее должны были размещаться на диске, теперь могут храниться непосредственно в ОЗУ. Это позволяет устранить многие препятствия, связанные с ограничениями производительности, что ранее казалось непреодолимым. Еще менее существенным ограничение производительности становится с появлением компьютеров с несколькими процессорами параллельных и конвейерных архитектур, ассоциативных процессоров. Таким образом, заложенное в традиционные СУБД предположение об обработке запроса единственным процессором перестает быть истинным.

Понимание того факта, что традиционные СУБД не способны решить новых возникших перед ними проблем, а технология ушла далеко вперед и перестала сдерживать программное обеспечение, привело к новому витку в развитии баз данных. Стали появляться прототипы СУБД нового поколения, рассчитанные на более универсальное использование. С другой стороны, разработчики пытались использовать новые технические средства, включая многопроцессорные компьютеры и компьютеры с большими объемами оперативной памяти, создавая машины баз данных.

Десять рассматриваемых в обзоре исследовательских проектов, перечислены в таблице 1 в алфавитном порядке с указанием организации, финансирующей проект и фамилии руководителя проекта. В таблице 2 эти системы классифицированы по их назначению, а в таблице 3 — по используемой в них модели данных. Таблица 2 показывает, что три системы ориентированы на повышение производительности работы, исследования которых направлены на использование оперативной памяти и параллельной обработки, т.е. представляют собой программное обеспечение машины баз данных. В одной системе внимание сосредоточено лишь на выполнении рекурсивных запросов, в то время как остальные функции СУБД авторами не рассматриваются. Наконец, остальные шесть систем являются универсальными СУБД, рассчитанными на широкую сферу приложений. Четыре из этих шести систем разрабатываются как обособленные СУБД, а две — являются надстройками к существующим пакетам. Второе направление классификации проектов, отраженное в таблице 3, показывает используемую в системах модель данных. Поскольку СУБД System M не имеет модели данных, в таблице отражены только девять оставшихся разработок. Три из них поддерживают традиционную реляционную модель данных, как и обычные коммерческие СУБД. Две системы используют реляционную модель с вложенностью, когда поле отношения, в свою очередь, может быть отношением. Наконец, оставшиеся четыре модели используют объектно-ориентированные модели данных, и включают однозначную идентификацию записей пользовательских функций и связи наследования между данными и функциями.Теперь остановимся на некоторых направлениях исследований, общих для рассматриваемых проектов. Можно выделить два направления. Первое — относится к области моделирования данных, в нем рассматриваются вопросы реализации в современных СУБД объектно-ориентированных моделей данных и продукционных правил. Второе направление исследований связано с построением новых аппаратно-программных архитектур СУБД, их расширяемости, включения новых функций обработки и создания версий, ориентированных на параллельную и распределенную обработку.

Таблица 1

Состав систем, рассматриваемых в обзоре

Система	Спонсор	Руководитель
BUBBA	Microelectronics and Computer Technology	Х. Борэл
DASDB	Дармштадский университет	Г. Шек
GAMMA	Висконсинский университет	Д. де Витт
IRIS	Лаборатории Hewlett Packard	П. Лингбэк
LDL	Microelectronics and Computer Technology	К. Заниоло
O2	ALTAIR	Ф. Бансильон
ORION	Microelectronics and Computer Technology	В. Ким
POSTGRES	Калифорнийский университет	М.Стоунбрэйкер
STARBURST	Исследовательский центр IBM	Л. Хаас
SYSTEM M	Принстонский университет	Г. Гарсиа-Молина

Таблица 2

Классификация систем по назначению

Программное обеспечение машины баз данных	Рекурсивные запросы	СУБД общего назначения, разработанные "с нуля"	СУБД общего назначения, разработанные в качестве надстройки
BUBBA	LDL	DASDB	IRIS
GAMMA		ORION	O2
SYSTEM M		POSTGRES	
		STARBURST	

Таблица 3

Классификация систем по используемой модели данных

Реляционная	Реляционная с вложенностью	Объектно-ориентированная
GAMMA	BUBBA	IRIS
LDL	DASDB	O2
STARBURST		ORION
		POSTGRES

Прежде всего остановимся на моделировании данных. Это направление подробно рассматривалось, например, при создании логического языка данных LDL, позволяющего поддерживать новому поколению баз данных быструю разработку таких сложных приложений, как экспертные системы и системы для решения научных и технических задач. Поставленная цель не нова, поскольку в области создания языков баз данных наблюдается повышенный интерес к их использованию в новых прикладных областях и установлению связей между базами данных и языками программирования. Однако большинство исследователей предпринимали попытки создания интерфейса между реляционными СУБД и традиционными языками. И только совсем недавно усилия стали направляться на интеграцию баз данных и языков программирования на основе объектно-ориентированной модели. Этот подход предполагает замену реляционных баз данных объектно-ориентированными, которые поддерживают более ограниченные языки запросов и способы навигации в базе данных, свойственные дореляционным системам. В противовес всем этим исследованиям в LDL принята точка зрения о максимальных возможностях программирования, которые могут быть достигнуты только за счет расширения реляционных языков запросов и методов, обеспечивающих интеграцию и эффективную поддержку такого расширенного языка в СУБД. Также предполагается, что система, полученная в результате этих исследований, будет представлять собой реализацию дедуктивной базы данных и явится важным пунктом на пути к созданию в будущем систем управления знаниями, которые должны сочетать в себе эффективный механизм логического вывода с надежными средствами управления большими информационными банками. В рамках проекта LDL, разработка которого началась в 1984 году, создан новый язык, новые методы компиляции и оптимизации запросов, а также эффективно работающий и компактный прототип системы.

Как и при создании логического языка LDL, целью проекта Starburst было построение системы управления реляционной базой данных, которая могла бы легко расширяться для поддержки приложений не типичных для реляционных систем, и использоваться в качестве исследовательского средства. Кроме того, авторы преследовали цель усовершенствовать структуру и алгоритмы традиционных систем и, тем самым, повысить их производительность.

Поставленные цели объясняются несколькими причинами. Авторы сочли лучшим способом обеспечения поддержки новых приложений СУБД — построение реляционной базы данных, которая могла бы легко расширяться, другой причиной является построение полной системы управления базой данных, включая

средства обеспечения параллельного управления, восстановления, авторизацию доступа, оптимизатор и процессор языка, а также совершенствование других функций базы данных.

Проект Starburst прошел две фазы: на первой фазе была реализована расширенная система управления реляционной базой данных с улучшенными характеристиками производительности, на второй фазе Starburst использовалась в качестве исследовательского средства, позволившего проанализировать возможности использования в СУБД пользовательских типов данных, сложных объектов, продукционных правил, хранения баз данных в оперативной памяти и параллельной обработки. К настоящему времени реализован прототип системы Starburst и продолжаются исследовательские работы.

В том же направлении шли работы и при реализации системы POSTGRES. Чтобы удовлетворить потребности прикладных областей, лежащих за пределами управления экономической информацией, ставилась задача расширить средства СУБД в двух направлениях: управлении объектами и управлении знаниями. Под управлением объектами понимается хранение и манипулирование данными нетрадиционных типов, например, геометрическими фигурами, пиктограммами, текстом и т.п. Объектно-ориентированные языки программирования и базы данных используются прежде всего в этих областях.

Управление знаниями предполагает способность хранить и использовать наборы продукционных правил, описывающих семантику прикладных областей. Такие правила определяют ограничения целостности данных базы данных, а также позволяют получать производные данные, не хранящиеся в базе.

Основной целью проекта POSTGRES было объединение в одной системе трех направлений: средства управления объектами и правилами были добавлены к средствам, типичным для традиционных СУБД.

Примером системы управления объектно-ориентированной базой данных является СУБД Iris, которая разработана в Hewlett-Packard Laboratories. Одной из целей этого проекта было повышение производительности труда проектировщиков баз данных за счет создания новой модели данных. Другой целью было обеспечение полной поддержки базы данных в процессе разработки и интеграции нетрадиционных приложений, включая инженерные информационные системы, тестирование и измерение в технике, телекоммуникации, автоматизацию учреждений, экспертные системы, а также проектирование аппаратного и программного обеспечения компьютеров.

Архитектура Iris включает ядро и процессор поиска и модификации данных. В ядре поддерживается объектно-функциональная модель данных: объекты модели Iris подразделяются по типам, но в отличие от других объектных систем, не имеют состояния; значения атрибутов, связи и поведение объектов представляются в виде функций. Поисковые и модифицирующие запросы записываются как функциональные выражения. Система расширяется за счет введения новых пользовательских функций. Функции могут быть реализованы в виде хранящихся в памяти таблиц или в виде вычисляемых выражений. Выражения записываются как на собственном языке Iris, так и на языке программирования общего назначения (например, Си).

Как и в большинстве других баз данных доступ к Iris возможен как посредством интерактивного интерфейса, так и через интерфейсы языков программирования. Все интерфейсы реализованы как клиенты ядра. Клиент форматирует запрос в виде выражения языка Iris и затем обращается к точке входа ядра, которая вычисляет выражение и возвращает клиенту результат, который также сформатирован в виде выражения Iris.

В настоящее время поддерживаются два интерактивных интерфейса. Один интерактивный интерфейс — это объектно-ориентированное расширение SQL (OSQL). Вторым интерактивным интерфейсом является графический редактор. Система построена на основе X-Windows и дает пользователям возможность выполнять поисковые и модифицирующие функции посредством графического интерфейса или средствами экранных форм.

Помимо интерфейса ядра Iris поддерживает еще два программных интерфейса. Первый, CLI (интерфейс языка Си) является пользовательской надстройкой к интерфейсу ядра. Он дает возможность программистам использовать объектно-ориентированные средства Iris при манипулировании с переменными Си, включая метаданные, функции и объекты. Второй программный интерфейс позволяет использовать в различных языках программирования OSQL.

Наконец, скажем несколько слов об объектно-ориентированной СУБД O2, разработанной в рамках проекта ALTAIR. Система поддерживает языки программирования базы данных CO2 и BasicO2, набор средств генерации интерфейса пользователя LOOKS и средства программной поддержки OOPE.

O2 состоит из следующего набора функциональных модулей: OOPE, алфавитно-цифрового интерфейса, LOOKS, языкового процессора, интерпретатора запросов, программы управления схемами, программы управления объектами и программы управления дисками.

Программа управления дисками отвечает за выполнение операций ввода-вывода, размещение, индексирование и буферизацию данных. Программа управления объектами обеспечивает отображение на дисковую память абстрактной объектной модели данных. Программа управления схемами работает со схемами, которые содержат в себе описание типов данных и программ. Языковый процессор обрабатывает команды языка описания данных и компилирует программы. Он также взаимодействует с программой управления схемами. Интерпретатор запросов отвечает за интерпретацию запросов с использованием программ управления объектами и схемами. LOOKS выполняет управление экраном и объектами изображе-

ния, взаимодействуя при этом с программой управления объектами. OOPE выполняет функции программной поддержки и взаимодействует с LOOKS при выводе и управлении данными на экране.

Рассмотрение проблемы построения новых архитектур СУБД мы начнем с двух СУБД, ориентированных на мультикомпьютерные системы, построенные на основе концепции shared-nothing, когда система может включать множество процессоров, соединенных между собой сетью в виде гиперкуба или кольца, а диски подключены непосредственно к процессорам. Общепризнано, что такие архитектуры могут включать до нескольких тысяч процессоров. Уже существует машина баз данных Teradata, которая включает более 200 процессоров.

Проект Bubba разрабатывается с 1984 года, его цель состоит в создании масштабной высоко производительной и доступной системы баз данных, обеспечивающей существенное превышение эффективности работы по отношению к традиционным универсальным машинам. Процесс проектирования выполнялся в несколько итераций, включая проектирование, моделирование и разработку прототипа. В настоящее время Bubba работает на мультикомпьютере, состоящем из 40 узлов и обеспечивает параллельную компиляцию, распределенное управление транзакциями, управление объектами данных и имеет собственную версию UNIX. В статье описывается существующий прототип системы и рассматриваются основные проектные решения, заложенные в него.

Технической основой для проекта Bubba послужила архитектура мультикомпьютера shared-nothing, способного включать в себя несколько тысяч узлов. Данные, хранящиеся в нем, декластеризованы, т.е. распределены по узлам вне зависимости от частоты их совместной обработки путем горизонтального секционирования, кэширования или т.п., а операции обработки выполняются в тех узлах, в которых и содержатся требуемые данные. В этом случае, по мнению авторов, достигается максимальный параллелизм как при выполнении отдельных транзакций, так и при функционировании множества параллельно выполняемых транзакций.

Цели проекта Bubba определили и основные направления исследовательских работ, которые должны были обеспечить эффективное управление параллельной обработкой. Исследования проводились по четырем направления:

Размещение данных — Bubba разрабатывалась как система для приложений с интенсивным обращением к данным, когда объемы данных настолько велики, а доступ к ним требуется столь часто, что обмен данными между узлами мультикомпьютера в процессе обработки по соображениям быстродействия невозможен. Операции выполняются по месту хранения данных. Вследствие этого декластеризация и размещение данных в узлах непосредственно влияют на степень загруженности системы. По мере изменения стуктуры и объема базы данных предусматривается соответствующее перемещение данных, обеспечивающее равномерность загрузки системы.

Автоматическое распараллеливание обработки — Одним из основных требований к проекту было обеспечение "прозрачности" мультикомпьютера, т.е. пользователь должен был лишь составить транзакцию для централизованной схемы базы данных, после чего компилятор Bubba автоматически декомпозировал транзакцию и размещал субтранзакции в различных узлах мультикомпьютера.

Управление потоками данных — В большинстве машин, управляющих потоками данных, операции каждого потока выполняются в одном узле. В Bubba каждая операция может выполняться параллельно несколькими процессорами. Узлы, участвующие в обработке, определяются данными, требуемыми для выполнения операции. Когда результаты выполнения операции передаются в узел, выполняющий следующую операцию, средства управления потоками данных должны указать узлу-получателю координаты передающих узлов и определить ориентировочное количество передаваемых сообщений. Эффективность управления потоками обеспечивается за счет идентификации передающих узлов, информирования узлов-получателей и минимизации перегрузок.

Методы восстановления данных — Bubba предназначен для областей, требующих высокой доступности данных. Тем не менее, с увеличением числа узлов возрастает и вероятность их отказов. В проекте разработан ряд методов, позволяющих системе сохранять работоспособность при отказе узлов и подключать резервные узлы взамен вышедших из строя.

В процессе выполнения проекта Bubba был реализован и ряд новых подходов в области баз данных.

При разработке системы Gamma на протяжении последних пяти лет основное внимание было сосредоточено на вопросах проектирования и реализации машин баз данных, обеспечивающих высокую степень параллелизма. Во многих отношениях разработка Gamma основывалась на опыте, приобретенном при создании его предшественника — машины баз данных DIRECT. Несмотря на то, что DIRECT обеспечивал определенную степень параллелизма, проект имел ряд серьезных недостатков, делающих невозможным расширять его архитектуру до нескольких сотен процессоров, главным образом, вследствие коллективного использования памяти и централизованного управления выполнением параллельных алгоритмов.

Для разрешения проблем, возникших в DIRECT, проект Gamma предлагает достаточно удобные решения. С точки зрения архитетуры, эта система основывается на концепции shared-nothing. Вторая ключевая идея, заложенная в Gamma, состоит в использовании параллельных алгоритмов кэширования. В отличие от DIRECT, в Gamma эти алгоритмы не требуют централизованного управления и могут расширяться вместе с аппаратными средствами. Наконец, в целях максимального использования ограниченной полосы пропускания каналов ввода-вывода современных

дисководов, в Gamma для распределения кортежей отношения среди множества дисководов используется концепция горизонтального секционирования или декластеризации. Этот подход позволяет обрабатывать большие отношения параллельно на нескольких процессорах, снижая тем самым перегрузки.

После завершения к концу 1984 года разработки программного обеспечения, началась работа над первым прототипом, который вступил в строй к концу 1985. Эта версия Gamma была выполнена в качестве надстройки к существующему мультикомпьютеру, состоящему из 20 процессоров VAX 11/750. За период 1986-1988 гг. в прототип были включены новые операции (операторы aggregate и update), новые методы параллельного соединения (Hybrid, Grace и Sort-Merge) и полностью параллельный механизм управления. Кроме того, был выполнен ряд исследований по производительности систем. К весне 1989 г. Gamma была установлена на гиперкубе, состоящем из 32 процессоров Intel iPSC/2.

Создание действительно "универсальной", т.е. на все случаи жизни, СУБД чрезвычайно сложно, да и не нужно. Решение проблемы универсальности предлагается в двух следующих проектах — DASDBS и ORION, эти системы построены как семейства, каждый из членов которого предназначен для решения вполне конкретных задач.

История системы баз данных Darmstadt, также известной как DASDBS, началась в 1983 году, однако эта система испытала на себе некоторое влияние более раннего проекта AIM, начатого в 1978 г. За это время возникли новые сферы приложения систем баз данных, что дало толчок разработкам в области СУБД. Как и многие исследователи, авторы этой системы считают, что усилия, направленные на включение дополнительных функций в контекст существующих традиционных баз данных в качестве надстройки, не могут принести желаемых результатов, поэтому работа над DASDBS велась "с нуля".

С другой стороны авторы проекта не пытались создавать СУБД на все случаи жизни. Напротив, вследствие разницы в требованиях к СУБД со стороны пользователей различных прикладных областей, DASDBS разрабатывалась как семейство систем баз данных, каждая из которых наиболее полно отражает требования одного из классов задач. Идея состояла в том, чтобы настраивать СУБД запуском сменяемой надстройки к универсальному ядру. Само ядро выполняло функции нижнего уровня, общие для задач управления данными различных классов задач.

В начале 1987 года прототип ядра DASDBS (написанный на ПАСКАЛе) начал функционировать в среде операционной системы VM/SP. К настоящему времени ядро адаптировано для работы под управлением Unix, оно значительно расширено, а внимание разработчиков сосредоточилось на создании прикладных надстроек.

Авторы проекта уделили большое внимание вопросам поддержки ненормализованных отношений, допускающих вложенность отношений друг в друга, использованию объектных буферов для обеспечения взаимодействия СУБД и компиляторов языков программирования, а также вопросы обработки многоуровневых транзакций.

Разработка проекта системы распределенной обработки данных ORION была начата в 1985 году. К настоящему времени в рамках проекта разработано и реализовано три системы баз данных: ORION-1, ORION-1SX и ORION-2. Эти СУБД предназначены для тех прикладных областей, в которых требуется использование объектно-ориентированных моделей данных: к числу этих областей можно отнести искусственный интеллект, автоматизированное проектирование и автоматизацию учреждений.

ORION-1 — однопользовательская многозадачная система.

ORION-1SX — система "клиент-сервер", разработанная для использования в локальных вычислительных сетях. Клиентами этой системы являются рабочие станции, не имеющие дисковой памяти, т.е. не имеют собственной базы данных. Сервер обеспечивает распределение базы данных и управляющей информации (например, таблиц блокировки и буфера постраничного обмена) среди клиентов. Клиенты поддерживают пользовательский интерфейс ORION, посредством которого прикладные программы могут использовать средства управления объектами ORION-1SX.

Третья СУБД этой серии, ORION-2, является распределенной системой управления объектами, в которой все компьютеры сети принимают участие в управлении данными коллективного пользования.

Системы ORION были реализованы на языке Common LISP, на LISP-машине Symbolics 3600, и в настоящее время адаптированы для использования в среде операционной системы Unix на рабочих станциях Sun. Таким образом, ORION-1 работает на Sun-3 или рабочей станции Symbolics, а ORION-1SX и ORION-2 функционируют в локальной сети рабочих станций Sun-3 или сети станций Symbolics.

Система ORION непосредственно поддерживает объектно-ориентированную модель и допускает коллективное использование объектов. Она также поддерживает ряд средств управления базой данных, особенно полезных в областях интенсивного использования данных, включая автоматическую оптимизацию запросов, управление транзакциями, динамическую модификацию схемы данных, средства поиска в тексте и т.п. Система ORION-2 будет включать также средства авторизации доступа, средства создания версий схемы базы данных и более обобщенную семантику представления составных объектов.

Окончание статьи на стр. 79

Практическое программирование на dBASE

3.3. Приемы программирования

В настоящее время существует достаточно много литературы по методам и технике программирования как в теоретическом плане, так и с практической ориентацией на определенные языки. Однако находящиеся в ней рекомендации имеют лишь общее отношение к dBASE. Дело здесь в том, что наиболее эффективный, простой и быстрый метод написания программы на dBASE — это максимальное использование универсальных средств автоматизации программирования, предоставляемых системой. Эти средства учитывают принятую в системе структуру хранения информации и поэтому работают достаточно эффективно. Кроме того, кождое стандартное средство (будь то команда EDIT или формат экрана) ориентировано на пользователя, т.е. является, как говорят, "дружественнным". Тем самым создаются предпосылки для написания программ, имеющих современные средства общения с пользователем. Использование стандартных средств dBASE значительно сокращает текст программы и время ее написания. Попутно появляется возможность унифицировать команды и способы взаимодействия с создаваемыми программами, так что пользователь, научившийся работать с одной из них, быстрее научится работать и с другими.

Чтобы показать связь между различными средствами языка, в конце этой главы представлены фрагменты системы ведения бибилиографической картотеки. Обычно описание требований, предъявляемых к системе, называют спецификацией или техническим заданием. В спецификации должно быть описано назначение системы, указаны ее функции, ограничения и приведены требования пользователя к производительности. В нашем случае нужна система, позволяющая работать с электронной картотекой. Каждая хранимая карточка должна иметь вид, представленный на рис. 2.3. Кроме того, должна быть предусмотрена возможность записывать в карточку дополнительную информацию любого характера (например, краткий реферат и т.п.). Эта информация записывается как бы на "обратной стороне" электронной карточки. Пользователь, работающий с карточкой, должен иметь следующие возможности:

выбрать картотеку для работы. Предполагается, что у него может быть много картотек. Каждая из них должна иметь имя;

осуществлять просмотр картотеки в порядке шифров карточек либо осуществлять поиск всех карточек, удовлетворяющих заданному критерию. Критерием отбора может быть любое поле карточки. Должны отыскиваться все карточки, в полях которых находится указанная информация;

во время просмотра найденных карточек необходимо отмечать их для вывода на печать или для удаления. Впоследствии отмеченные карточки могут быть распечатаны в виде списка найденной литературы или непосредственно в виде карточек;

должна быть предусмотрена возможность физически удалять отмеченные для удаления карточки, освобождая место на внешнем носителе для новых карточек;

для первоначального ввода информации должен быть предусмотрен режим ввода с возможностью корректировки. Кроме того, для исправления допущенных ошибок или введения дополнительной информации должен быть предусмотрен режим корректировки уже существующих карточек;

чтобы пользователю картотеки не приходилось обращаться к операционной системе, должны быть предусмотрены функции создания, переименования, удаления и перечисления имеющихся картотек.

В спецификации обычно указываются требования к объемам информации, ограничения на время взаимодействия с пользователем и т.д. Мы же ограничимся тем, что наша система должна работать в среде пакета dBASE III plus и удовлетворять требованиям операционных систем MSDOS или PC DOS.

3.3.1. Программирование меню

Современные системы, предназначенные для пользователей-непрофессионалов, обычно работают под управлением меню. Под меню мы будем понимать со-

---

Главы из книги Д.Г.Берещанского "Практическое программирование на dBASE", Финансы и статистика, М., 1989 (Продолжение. Начало в №№ 7-10).

ставленную заранее и предназначенную пользователю систему выборов. В альтернативных меню можно выбрать только один любой пункт из представленного множества. В наборных меню можно выбрать сразу несколько из предложенных пунктов. Кроме того, пункты меню могут располагаться на экране вертикально или горизонтально. Для их выбора можно применять курсор, указатель типа "мышь" или световое перо, или вводить номер пункта либо соответствующую ему букву. Однако в dBASE наиболее просто программируются так называемые перекрывающие вертикальные меню с альтернативным выбором. Основной их отличительной особенностью является то, что меню, появляющееся на экране, перекрывает ранее находившуюся там информацию. Поэтому для этих целей идеально подходят команды **TEXT, ENDTEXT,** и **DO CASE**. Такие меню удобно применять в начале работы или в те моменты, когда какая-то операция завершена и нужно приступить к следующей. Ниже приведено основное меню библиографической картотеки:

```
           Основное меню каротеки
   ____________________________________

1. Создание/Выбор картотеки
2. Поиск/Просмотр карточек
3. Вывод на печать карточек или списков
4. Добавление новых карточек
5. Редактирование карточек
6. Манипуляции с карточками
7. Конец работы
Введите номер выбранного пункта = = = = = >
```

Как видим, оно охватывает все запросы пользователя, указанные в спецификации, и представляет собой первый кадр сценария работы с системой. Внизу экрана приводится краткое руководство пользователю по действию в данной ситуации — "местная подсказка". Ее наличие очень удобно на начальном этапе работы системы, однако она становится излишней после того, как пользователь научится работать. Поэтому наиболее подходящим видом помощи пользователю является вызываемая подсказка. Она появляется на экране при нажатии на определенную клавишу и исчезает при повторном нажатии. Очень часто для этих целей используют функциональную клавишу F1. Для реализации вызываемой подсказки можно использовать оператор ON KEY, о котором рассказывалось в 3.2.3. Иногда вызываемая подсказка имеет внешний вид, подобный меню, но это чаще всего просто список возможностей, предоставленных в данной ситуации. Выбор одной из этих возможностей и взаимодействие с системой осуществляются не через меню подсказки, а с помощью доступных в данной ситуации команд. Примером такого использования подсказки может служить системная подсказка для команд типа BROWSE, EDIT и т.п. Покажем далее, как программируется простое меню, посредством которого пользователь взаимодействует с системой. Ниже приведена программа, реализующая основное меню предлагаемой библиографической системы.

```
SET TALK OFF
CLEAR
TEXT
                Основное меню картотеки
   ____________________________________

1. Создание/Выбор картотеки
2. Поиск/Просмотр карточек
3. Вывод на печать карточек или списков
4. Добавление новых карточек
5. Редактирование карточек
6. Манипуляции с карточками
7. Конец работы
ENDTEXT
@ 18,5 SAY""
STORE 7 TO Soobl
WAIT " Введите номер выбранного пункта > " TO Soobl
```

В программе оператор WAIT воспринимает любой символ, введенный с клавиатуры, поэтому после него необходима проверка правильности введенной информации. В случае ошибки пользователю должно быть выдано предупреждение. Для проверки воспользуемся оператором DO CASE.

```
DO CASE
  CASE Vybor = "1"
    DO KartoUs
  CASE Vybor = "2"
    DO KartoPo
  CASE Vybor = "3"
    DO KartoPe
  CASE Vybor = "4"
    DO KartoDo
  CASE Vybor = "5"
    DO KartoRe
  CASE Vybor = "6"
    DO KartoVy
  CASE Vybor = "7"
    EXIT
OTHERWISE
    STORE " Вы ошиблись, введите;
          любую цифру от 1 до 7 = = = > " TO Soobl
ENDCASE
```

Если объединить приведенные выше программы, то получится программа, вызывающая на экран основное меню и воспринимающая ответ пользователя. Но она выполнит эти действия только один раз. Чтобы повторно сделать выбор, придется вызывать программу вновь. С этой целью воспользуемся командой организации цикла — DO WHILE. Чтобы цикл заканчивался только по требованию пользователя, возьмем в качестве условия его повторения всегда истинное выражение:

```
DO WHILE .T.
<тело цикла>
ENDDO
```

Для выхода из цикла DO WHILE, когда пользователь выбрал пункт "КОНЕЦ РАБОТЫ", воспользуемся

оператором EXIT. Изменим также способ выдачи подсказки. Для этого введем переменную Soobl, вначале принимающую значение “Введите номер ...“, а после ввода неверного знака принимающую значение “Вы ошиблись ...“. Окончательный вариант программы вывода основного меню показан на рис. 3.4.

```
*Секция установок и присвоений
*
SET TALK OFF
STORE “ Введите номер выбранного пункта > “ TO Soobl
*
*Основная секция
*
DO WHILE .T.
clear
TEXT
                    Основное меню картотеки
          ____________________________________________

1. Создание/Выбор картотеки
2. Поиск/Просмотр карточек
3. Вывод на печать карточек или списков
4. Добавление новых карточек
5. Редактирование карточек
6. Манипуляции с карточками
7. Конец работы
ENDTEXT
@ 18,5 SAY““
STORE “7“ TO Vybor
WAIT Soobl TO Vybor
STORE “ Введите номер выбранного пункта > “ TO Soobl
DO CASE
 CASE Vybor = “1“
  DO KartoUs
 CASE Vybor = “2“
  DO KartoPo
 CASE Vybor = “3“
  DO KartoPe
 CASE Vybor = “4“
  DO KartoDo
 CASE Vybor = “5“
  DO KartoRe
 CASE Vybor = “6“
  DO KartoVy
 CASE Vybor = “7“
  EXIT
OTHERWISE
  STORE “ Вы ошиблись, введите любую цифру;
      от 1 до 7 = = = > “ TO Soobl
ENDCASE
ENDDO
*
* Секция завершения
*
CLOSE ALL
```

Рис. 3.4

В ней предусмотрены: секция присвоения начальных значений и установки системных параметров, основная секция и секция, выполняемая при окончании программы. Поскольку нас в данный момент интересует только основная часть программы, в остальных секциях предусмотрены лишь необходимые в этой ситуации команды. Остальные команды будут добавляться и вноситься в эти секции по мере развития программы.

В нашем примере для вывода текста вертикального меню применялась конструкция **TEXT/ENDTEXT**. В более сложных случаях, когда на экране должны использоваться поля повышенной яркости, разных цветов и т.д., более удобным является применение команд ? и @. Например, программа, реализующая основное меню бибилиографической системы, но с выделением первых букв пунктов по яркости и ведущих букв по цвету, показана на рис. 3.5. Как видим, в этом случае мы вынуждены воспользоваться командой (@) для позиционирования текста на экране и командой **SET COLOR TO** для выделения отдельных слов и букв.

Вертикальные меню создаются в dBASE достаточно удобно. Их легко программировать и они обычно удовлетворяют большинство пользователей. Однако в последнее время в современных системах широко стали применяться так называемые горизонтальные меню, занимающие гораздо меньше места и оставляющие почти весь экран под обрабатываемую в данный момент информацию. Как уже отмечалось, вертикальные меню перекрывают информацию пользователя на экране и это создает некоторый дискомфорт. Поэтому горизонтальные или неперекрывающие меню обычно используются в том случае, когда пользователь желает во время работы с системой постоянно видеть возможные варианты обработки информации или другие возможности дальнейшей работы.

```
STORE “Введите подсвеченную букву “ TO Soobl
CLEAR
DO WHILE.T.
SET COLOR TO W + /N
@ 1,20 SAY “Основное меню картотеки“
@ 2,17 SAY REPLICATE(CHR(205),30)
SET COLOR TO W + /N
@ 4,5 SAY “С“
SET COLOR TO W/N
?? “оздание/Выбор картотеки“
SET COLOR TO W + /N
@ 6,5 SAY “П“
SET COLOR TO W/N
?? “оиск/Просмотр карточек“
SET COLOR TO W + /N
@ 8,5 SAY “В“
SET COLOR TO W/N
?? “ывод на печать карточек или списков“
SET COLOR TO W + /N
@ 10,5 SAY “Д“
SET COLOR TO W/N
?? “обавление новых карточек“
SET COLOR TO W + /N
@ 12,5 SAY “Р“
SET COLOR TO W/N
```

```
?? "едактирование карточек"
SET COLOR TO W+/N
@ 14,5 SAY "М"
SET COLOR TO W/N
?? "анипуляции с карточками"
SET COLOR TO W+/N
@ 16,5 SAY "К"
SET COLOR TO W/N
?? "онец работы"
SET COLOR TO
@ 18,5
STORE "К" TO Vybor
WAIT Soobl TO Vybor
STORE "Введите подсвеченную букву " TO Soobl
DO CASE
 CASE Vybor$"Сс"
  DO KartoUs
CASE Vybor$"Пп"
  DO KartoPo
CASE Vybor$"Вв"
  DO KartoPe
CASE Vybor$"Дд"
  DO KartoDo
CASE Vybor$"Рр"
  DO KartoRe
CASE Vybor$"Мм"
  DO KartoVy
CASE Vybor$"Кк"
  EXIT
OTHERWISE
  STORE " Вы ошиблись! " + Soobl TO Soobl
ENDCASE
ENDDO
```

Рис. 3.5

Если "дерево" возможных вариантов велико, можно использовать двухуровневое горизонтальное меню. На первом уровне (обычно в верхней строке) перечисляются все выборы, доступные в данный момент, а на втором уровне (обычно следующая строка) — подвыборы текущего пункта первого уровня. В более сложных случаях можно комбинировать горизонтальные и вертикальные меню, как это сделано в команде ASSIST. В нашем примере горизонтальное меню можно было бы применить вместо главного вертикального. Поскольку в нем 7 пунктов, на каждый придется в среднем [80:7] = 11 позиций в строке. Учитывая хотя бы по одному пробелу-разделителю между пунктами, на один выбор придется всего 10 позиций. Необходимо придумать сокращения для названий каждого пункта. Например, вместо "Редактирование карточек" нужно написать просто "Редактирование" и т.п. Для пояснения сокращенных названий пунктов можно использовать вторую строку, в которой при движении курсора по пунктам меню первого уровня будет появляться их краткое описание. Как видим, приведенный вариант меню не имеет явных преимуществ по сравнению с вертикальным, поскольку не улучшает взаимодействия с пользователем, а, наоборот, даже несколько ухудшает его. Поэтому применять горизонтальное меню нужно в тех случаях, когда оно на самом деле необходимо. Например, при просмотре карточек в то время, когда на экране находится текущая карточка, пользователь может: просмотреть следующую, вернуться к просмотру предыдущей, закончить просмотр и вернуться в основное меню, либо приступить к поиску информации по шаблону. Все эти выборы можно расположить на одной строке, а на второй дать им пояснение. Для демонстрации возможностей языка dBASE приведем пример реализации такого меню (рис. 3.6). Поскольку не на всех системах имеются цветные мониторы, для выделения текущего пункта меню используется повышенная интенсивность основного цвета.

```
i = 1
max = 7
STORE "Следующая" TO Punkt1
STORE "Показ следующей карточки" TO Help1
STORE "Предыдущая" TO Punkt2
STORE "Показ предыдующей карточки" TO Help2
STORE "Печать" TO Punkt3
STORE "Ометить карточку для вывода на печать";
     TO Help3
STORE "Удаление" TO Punkt4
STORE "Ометить карточку для удаления" TO Help4
STORE "Шаблон" TO Punkt5
STORE "Поиск по шаблону" TO Help5
STORE "Отменить" TO Punkt6
STORE "Снять отметку печати или удаления" TO Help6
STORE "Возврат" TO Punkt7
STORE "Возврат в основное меню" TO Help7
 DO WHILE .T.
  STORE "Help" + LTRIM(STR(i,2,0)) TO Help
  j = 1
  @ 20,1      && Очистка строки с пунктами меню
  DO WHILE j< = max
    STORE "Punkt" + LTRIM(STR(j,2,0)) TO Punkt
  IF j = i
  SET COLOR TO w + /G
  ?? &Punkt       && Выделить только текст
  SET COLOR TO
  ?? SPACE (2)      && Пробелы фонового цвета
ELSE
?? &Punkt + SPACE (2)
ENDIF
j = j + 1
ENDDO     && Конец цикла вывода меню
 @ 21,1 && Очистить строку Help-a
 @ 21,1 SAY &Help
* Ввод нажатия с клавиатуры
 key = 0
DO WHILE key = 0
key = INKEY()
ENDDO
* Реакция на нажатие
 DO CASE
  CASE key = 27 && Esc
   EXIT
```

```
    CASE key = 13
      DO Karto Vy WITH i
    CASE key = 4
      i = IIF(i < max,i + 1,i)
    CASE key = 19
      i = IIF(i > 1,i-1,i)
  ENDCASE
ENDDO       && Конец главного цикла вывода меню
RETURN
```

Рис. 3.6

Таким образом, рассмотрены основные способы программирования меню для общения с пользователем в системе dBASE. Однако рассмотренные меню являются лишь наиболее доступными и простыми способами общения с пользователем. Современные системы предъявляют к пользовательскому интерфейсу значительно более высокие требования. Например, вместо названий пунктов меню требуется использовать пиктограммы (графические изображения объектов), а вместо управления курсором с помощью клавишей использовать указатель типа "мышь". Для реализации таких требований в dBASE предусмотрена возможность выполнения подпрограмм, написанных на другом языке программирования, например, СИ или Макроассемблере. Для этого нужно создать с помощью средств операционной системы двоичный файл (расширение имени .BIN). В операционной системе для этих целей служит программа EXE2BIN. Подпрограмму нужно писать как повторно используемую, т.е. такую, которую можно использовать без повторной загрузки с диска. Нельзя использовать память вне модуля. Общая длина не должна превышать 3200 байт. Подпрограмме может быть передан параметр в операторе

CALL <имя модуля>
  [WITH <выражение/переменная>]

Он может быть выражением символьного типа либо переменной. Выражения и переменные символьного типа должны заканчиваться байтом, состоящим из двоичных нулей.

При вызове подпрограммы регистр CS указывает на начало модуля, а регистры DC и BX — на первый байт параметра. Оператор CALL обращается к двоичному модулю, предварительно загруженному с помощью команды

LOAD <имя.bin-файла>

Таких файлов одновременно может быть загружено до 16. Если общее число двоичных модулей больше 16, то для загрузки новых модулей придется освобождать ненужные в данный момент подпрограммы командой

RELEASE MODULE <имя модуля>

Для трансляции такой программы нужно воспользоваться транслятором с Макроассемблера. Для этого достаточно в командной строке операционной системы написать:

MASM Asmprog,Asmprog

После этого можно вызвать редактор командой

LINK Asmprog,Asmprog

и превратить EXE модуль в двоичный файл оператором

EXE2BIN Asmprog

Для вызова подпрограммы в dBASE надо выполнить следующие команды:

```
LOAD Asmprog
CALL JIMMY WITH Param1
```

3.3.2. Создание и редактирование баз данных

В 2.2.1 была описана команда CREATE, с помощью которой можно в диалоговом режиме описать структуру и создать базу данных. Однако такой метод создания базы не применим, например, тогда, когда нужна временная база данных для хранения промежуточных результатов. В этом случае пользователь ничего не знает о структуре базы и не сможет создать ее в режиме диалога с командой CREATE. Конечно, можно создать все временные базы заранее, описав их структуры и оставив пустыми. Но в этом случае понадобится дополнительная внешняя память и, кроме того, придется описать их в инструкции пользователю, чтобы он представлял назначение каждой такой базы и случайно ее не уничтожил. Ведь в случае исчезновения одной из таких баз вся система перестанет правильно функционировать. Чтобы помочь в такой или подобной ситуации, dBASE имеет несколько возможностей создания базы данных на основе другой базы или на основе ее описания. Для этого можно использовать команду:

CREATE <новая база>
  FROM <структурный файл>

<структурный файл> — это файл, созданный командой

COPY TO <структурный файл>
  STRUCTURE EXTENDED

Как уже было описано в 2.2.1, этот файл является базой данных (.dbf) с жестко заданной структурой. Каждая его запись описывает одно поле копируемой базы данных и сама состоит из четырех полей:

FIELD-NAME — содержит имя описываемого поля;
FIELD-TYPE — содержит его тип;
FIELD LEN — содержит его длину;
FIELD-DEC — содержит количество десятичных позиций.

В качестве примера применения этих операторов приведем следующую программу:

```
* Модуль картотеки Kartous
STORE "?" + space(7) to Usname
DO WHILE Usname = "?"
  @ 20,1 CLEAR
  @ 20,1 SAY "Введите имя картотеки или ?";
     GET Usname PICTURE "@!"
  READ
  IF Usname = "?"
    CLEAR
    DIR
```

```
    ELSE
     Usname = TRIM(Usname) + ".dbf"
     IF file (Usname)
      USE &Usname
     ELSE
      CREATE &Usname FROM Karto.str
     ENDIF
    ENDIF
  ENDDO
  RETURN
```

Это первый модуль картотеки, описанной в 2.2.2. Он вызывается из основного меню картотеки и предназначен для выбора существующей или создания новой базы данных. Поскольку все картотеки имеют одинаковую структуру, их можно создавать командой CREATE FROM. Описание структуры хранится в файле Karto.str. Так как файл с описанием структуры сам является базой данных, с его записями можно выполнять различные операции. Можно, например, изменяя названия полей и их параметры, синтезировать описания других баз и затем создавать их командой CREATE FROM. Это свойство dBASE особенно полезно в тех применениях, где структура некоторой базы может быть заранее вообще не известна, и должна синтезироваться, исходя из поступающей информации. Но даже в тех случаях, когда мы заранее знаем структуру базы, можно применить синтез для того, чтобы избежать утомительного ввода о`днообразной информации в команде CREATE. Для иллюстрации приведем пример.

Допустим, мы создаем базу данных для ввода информации анкетного опроса. Если, например, в анкете 70 вопросов (а это может быть заранее не известно и выяснено во время диалога с пользователем), на каждый из которых пользователь может дать от 1 до 10 ответов, пронумерованных цифрами от 0 до 9, то структура базы данных может выглядеть, как в табл. 3.2.

Таблица 3.2.

Имя поля	Тип	Длина	Дес.поз.
Nomer	N	4	0
V1	N	1	0
V2	N	1	0
......			....
V70	N	1	0

Из структуры видно, что запись базы данных состоит из 71 поля. Первое поле названо NOMER, и в него будет записан номер анкеты. Далее поля V1...V70 представляют номера вопросов, и в них содержатся цифровые коды ответов на эти вопросы. Ввод информации о структуре базы данных представляет утомительную и однообразную работу, поскольку имена полей отличаются лишь индексами, а количество вводимых полей велико. Когда же число вопросов заранее не известно и задается пользователем в диалоге с программой, такую базу создать заранее вообще невозможно. В базах данных, предназначенных для накопления информации из анкет, подобные проблемы могут быть решены, например, с помощью следующей программы:

```
  SET TALK off
  STORE 1 TO i
  STORE 0 TO Fn
  ACCEPT "Введите имя новой базы данных:" TO Name
  STORE "& Name" + ".dbf" TO Name
  IF file ("&Name")
    ERASE &Name
  ENDIF
  INPUT "Введите число полей" TO Fn
  USE Template
  COPY STRUCTURE TO Temp
  USE Temp
  APPEND BLANK
  REPLACE FIELD_NAME WITH "Nomer"
  REPLACE FIELD_TYPE WITH "N"
  REPLACE FIELD_LEN WITH 5
  REPLACE FIELD_DEC WITH 0
  DO WHILE i< = Fn
   APPEND BLANK
   REPLACE FIELD_NAME WITH "V" + LTRIM(STR(i,3,0))
   REPLACE FIELD_TYPE WITH "N"
   REPLACE FIELD_LEN WITH 2
   REPLACE FIELD_DEC WITH 0
   i = i + 1
  ENDDO
  CREATE &Name FROM Temp
  USE
  ERASE Temp.dbf
```

Идея программы состоит в том, чтобы представить анкету в виде вектора V[i], размер которого может быть задан пользователем. Каждый элемент вектора соответствует полю в базе данных, поэтому все поля начинаются с буквы V, за которой следует номер поля. Ясно, что номера полей соответствуют номерам вопросов анкеты. Поскольку в команде CREATE STRUCTURE нельзя задавать имена полей в виде вектора, воспользуемся для создания базы данных командой CREATE FROM.

Как видно из примера, после заполнения информацией вспомогательной базы Temp специальной командой

```
CREATE &Name FROM Temp
```

создается новая база данных, имя которой указано в поле &Name, а структура описана в базе данных Temp. В данном случае нужна только одна специальная (может быть пустая) база Template (шаблон), в описании структуры которой заданы все четыре поля: FIELD_NAME, FIELD_TYPE, FIELD_LEN и FIELD_DEC. Чтобы не изменять шаблон, на его основе командой:

```
COPY STRUCTURE TO Temp
```

создается база данных Temp с описанием всех полей создаваемой базы данных. Поскольку база Temp вначале пуста, то для добавления в нее записей используется команда APPEND. Операнд BLANK позволяет ввести в базу пустую запись, не вступая в диалог с

пользователем. Для заполнения этой записи используется команда:

REPLACE [<диапазон>]<поле>
WITH <выражение>
[,<поле> WITH <выражение>]....
[WHILE <условие>] [FOR <условие>]

Если операнд <диапазон> не задан, то изменяется только заданное поле текущей записи в активной в данный момент базе данных. После того, как описание базы данных будет синтезировано, с помощью команды CREATE FROM создается и сама база данных.

Большие базы данных создаются обычно по частям. Вначале накапливается группа записей. Она проверяется, сортируется, обрабатывается. На ее основе создаются текущие отчеты. Только после этого новая информация помещается в основную базу данных, которая служит как бы хранилищем всей информации и источником получения сводных отчетов. Если накопленная информация должна быть просто добавлена в конец существующей базы данных, наиболее удобно сделать это с помощью команды:

APPEND FROM <имя файла>
[FOR <условие>] [[TYPE] <тип файла>]

Эта команда всегда добавляет данные в конец базы. Поэтому для сохранения упорядоченности необходима предварительная сортировка добавляемых записей. Операнд TYPE задает тип добавляемого файла. Этот операнд был описан в 2.2.5, где команда использовалась для ввода данных, подготовленных в других системах. Здесь же мы используем ее для слияния накопленной информации с информацией, хранимой в большой базе данных. В этом случае операнд TYPE не указывается. При слиянии команда APPEND учитывает только поля, имеющие одинаковые имена в обеих базах. Таким образом, вспомогательная база может иметь структуру, отличную от структуры основной базы.

Встречается при обработке данных и обратный процесс. Часто база данных создается программным путем потому, что необходимо отобрать из уже существующей большой базы только те записи, которые будут нужны для дальнейшей обработки. Такой метод позволяет существенно ускорить работу обрабатывающих программ, так как после отбора эти программы каждый раз будут просматривать значительно меньше записей. Иногда это вообще единственно возможный метод, как, например, в том случае, когда на внешнем носителе недостаточно пространства для хранения всей базы. Тогда необходимо отбирать для работы только нужную информацию. Это можно делать на большем компьютере, а обработку вести на меньшем. Для таких целей в языке имеется команда:

COPY TO <новый файл> [<диапазон>]
[FIELDS <список полей>]
[FOR <условие>] [WHILE <условие>]
[TYPE <тип файла>]

Команда эта была подробно описана в 2.2.5, однако там рассматривалась ее способность отбирать записи для копирования, удовлетворяющие заданным условиям. Условия эти задаются в операндах WHILE или FOR. Например, если пользователя интересуют только записи последнего дня, можно задать

COPY TO Lastday ALL FOR Rec_date = ctod("31/07/87")

и получить базу Lastday, содержащую только записи, введенные 31 июля 1987 года. В дальнейшем, с новой базой можно производить различные манипуляции: сортировать, группировать, пополнять, изменять или получать сводные результаты.

Иногда необходимо создать новую базу на основе информации, хранимой в других, уже существующих базах. Поскольку dBASE позволяет работать одновременно с несколькими базами, открытыми в разных рабочих областях, создание новой базы путем слияния отдельных полей двух других баз нужно использовать только в тех случаях, когда это действительно необходимо. Например, если новая база будет значительно меньше за счет сокращения длины записи и отбора только необходимых записей. Тогда ее можно будет обрабатывать на компьютере с ограниченным объемом оперативной памяти, поскольку на таком компьютере нельзя иметь много одновременно открытых файлов. Этот способ подходит и для компьютеров с ограниченным пространством внешней памяти, например, не имеющих твердых дисков, так как на них нельзя обрабатывать большие файлы. Для слияния баз данных в этом случае используется команда:

JOIN WITH <альтернативное имя>
TO <новая база> FOR <условие>
[FIELDS <список полей>]

С ее помощью для каждой записи из активной базы данных находятся подходящие записи в альтернативной базе, открытой в другой зоне и указанной в операнде WITH. В новую запись входят все поля из обеих баз, указанные в FIELDS. С каждой записью из активной базы сливаются все записи из альтернативной базы, удовлетворяющие условию, указанному в операнде FOR. Если условие в операнде FOR указано неправильно, то результативная база данных может получиться очень большой. Поэтому при выборе условия FOR нужно быть особенно внимательным. Например, база Spisok является справочником деталей и имеет структуру, показанную в табл. 3.3.

Таблица 3.3

Имя поля	Тип	Длина	Дес.поз.
Nazvanije	C	40	
Cena	N	6	2
Kod	C	4	

В базе данных Detali хранится информация о количестве деталей определенного типа, произведенных за день (табл. 3.4).

Таблица 3.4

Имя поля	Тип	Длина	Дес.поз.
Kod	C	40	
Kolicestvo	N	6	2
Data	C	4	

Для печати отчета нужно слить эти базы таким образом, чтобы в записях присутствовали поля из обеих баз. Этого можно добиться командами:

```
SELECT 2
USE Detali
SELECT 1
USE Spisok
JOIN WITH B TO Otciot FOR Kod = B->Kod;
FIELDS Kod,Nazvanije,Cena,B->Kolicestvo,B->Data
```

Существует и еще один метод создания баз данных — это команда

IMPORT FROM <имя файла> [TYPE] PFS

Как уже упоминалось выше, dBASE позволяет вместе с файлом базы данных сохранять информацию об используемых для работы с этой базой форматных файлах и связях с другими базами. Вся эта информация хранится в специальных Pfs-файлах. Команда IMPORT, основываясь на информации, хранимой в Pfs-файле, восстанавливает базу данных и другие связанные с ней файлы. Наиболее удобно использовать эту команду в том случае, когда необходим обмен информацией на уровне файлов между двумя задачами, написанными на dBASE. Тогда одна из задач с помощью команды

EXPORT TO <имя файла> [TYPE] PFS

подготавливает всю необходимую информацию в Pfs-файле, имя которого задается в операнде TO. Другая задача получает эту информацию с помощью команды IMPORT. Если же хранимые в базе данные должны обрабатываться программами, написанными на других языках, то с помощью команды

COPY TO <имя файла>... TYPE <тип файла>

можно создать файл необходимого типа (смотрите описание команды в 2.2.1).

Мы уже знаем, что в dBASE есть мощные средства редактирования записей: команды EDIT/CHANGE и BROWSE. Однако их основным недостатком является то, что изменения вносятся непосредственно в базу данных без предварительного контроля. Использование форматных файлов в значительной мере ослабляет этот недостаток за счет использования операндов PICTURE и RANGE в операторе @. Эти же средства, разумеется, можно применять и при использовании команды @ в программах. Например, команды:

```
@ 10,15 SAY "Введите число";
 GET Number PICTURE "99";
 RANGE 0,25
 READ
```

позволяют ввести число и проверить, попадает ли оно в допустимый диапазон. В некоторых случаях нужны более сложные проверки, например, когда во вводимой информации используются контрольные цифры. Поэтому в языке существует команда, позволяющая вносить изменения в записи базы данных после выполнения всех необходимых проверок. Это рассмотренная ранее команда **REPLACE**. С ее помощью можно вносить изменения в текущую запись или во все записи, удовлетворяющие условиям WHILE и FOR. Если изменению подвергается поле, по которому база данных проиндексирована, то автоматически будет изменен и соответствующий индекс. Например, команда:

```
REPLACE ALL Kolicestvo WITH Kolicestvo-1;
FOR Data = ctod("31/07/87")
```

уменьшит на единицу значение в поле Kolicestvo во всех записях, созданных 31 июля 1987 года.

На практике часто возникает задача корректировки записей главной базы данных по информации, хранимой во вспомогательной базе. В этом случае условием корректировки записей в главной базе является присутствие соответствующей записи во вспомогательной базе. Такая задача является стандартной для процедуры ведения складского хозяйства. Например, дополним записи описанной ранее базы Spisok цифровым полем Vsego, в котором будет накапливаться общее количество произведенных деталей. Тогда на основе информации о произведенных в течение дня деталях, накопленной в базе Detali, можно будет в конце дня обновить информацию в базе Spisok. С этой целью можно применить команду:

```
UPDATE ON <ключевое поле>
       FROM <альтернативное имя>
       REPLACE <поле>
       WITH <выражение>,[<поле>
       WITH <выражение>,...] [RANDOM]
```

Эта команда изменяет значения полей, перечисленных в операнде REPLACE, на значения указанных выражений. Замена производится только в тех записях текущей базы, ключевое поле которых совпадает со значением поля из альтернативной базы данных, указанного в операнде ON. Альтернативная база должна быть открыта в другой области. Лучше, если обе базы будут отсортированы или проиндексированы. В том случае, когда альтернативная база не упорядочена, нужно указывать операнд RANDOM. Если основная база имеет множественные записи, то изменена будет только первая из них. Это позволит получить правильную суммарную информацию в дальнейшем.

Программа, обновляющая основную базу на основе файла накопленных данных, и исходное состояние баз показано на рис. 3.7.

В качестве второго примера применения команды UPDATE приведем систему, состоящую из главной базы (Master-base) и нескольких вспомогательных (операционных) баз. В операционных базах или, как их иногда называют, "транзакциях", накапливаются записи об операциях текущего периода (час, день, неделя и т.д.).

```
SET TALK OFF
SELECT B
USE Detali
LIST ALL
SELECT A
USE Spisok INDEX Spisok
LIST ALL
UPDATE ON Kod FROM Detali RANDOM;
  REPLACE Vsego WITH Vsego + B->Kolicestvo
LIST ALL
```

База данных Detali

Record #	KOD	KOLICESTVO	DATA
1	01-1	20	01/07/87
2	01-2	12	01/07/87
3	01-3	6	01/07/87
4	1715	100	02/07/87

База данных Spisok до команды UPDATE ON

Record #	NAZVANIJE	CENA	KOD	VSEGO
1	Деталь 1	100.00	01-1	20.00
2	Деталь 2	45.90	01-2	12.00
3	Деталь 3	12.00	01-3	6.00
4	Роботрон 1715	500.00	1715	101.00

База данных Spisok после команды UPDATE ON

Record #	NAZVANIJE	CENA	KOD	VSEGO
1	Деталь 1	100.00	01-1	40.00
2	Деталь 2	45.90	01-2	24.00
3	Деталь 3	12.00	01-3	12.00
4	Роботрон 1715	500.00	1715	201.00

Рис. 3.7

После соответствующей проверки и печати отчетов о текущем периоде эти записи используются для корректировки информации в главной базе (Master-base). Например, в операционных базах хранятся записи о поступлении товара и его выдаче со склада. При этом в базе Postupil фиксируется только поступление, а в базе Vydan — только выдача. В конце рабочего дня необходимо на основе этой информации обновить Master-файл Nalicije. С этой целью для каждой записи в базе Postupil находят соответствующую запись в файле Nalicije и складывают имеющееся в наличии количество с поступившим количеством. После этого для каждой записи в файле Vydan выполняют аналогичную операцию, но со знаком минус. В результате этих изменений основной файл правильно отражает наличие товара на складе.

Таблица 3.5

Имя поля	Тип	Длина	Дес.поз.
Kod	N	6	0
Nazvanije	C	40	
Kolicestvo	N	4	0
Cena	N	6	2

Таблица 3.6

Имя поля	Тип	Длина	Дес.поз.
Kod	N	6	0
Postupilo	N	4	0
Data	D	8	
Postavscik	C	20	

Таблица 3.7

Имя поля	Тип	Длина	Дес.поз.
Kod	N	6	0
Vydano	N	4	0
Data	D	8	
Polucil	C	20	

Приведенный метод — "главный файл — файлы транзакции" — позволяет перед обновлением главной базы произвести необходимые проверки вводимой информации и получить отчеты о текущих операциях. Это большой плюс системы, поскольку файлы транзакции значительно меньше основного и обрабатываются достаточно быстро. Кроме того, предварительная проверка информации перед обновлением базы позволяет избежать множества ошибок. Если после обновления Master-файла все же окажется, что информация введена неверно, то всегда можно повторить операцию обновления после исправления ошибок в файлах транзакций, используя копию основного файла. Выше приведены таблицы 3.5, 3.6, 3.7 со структурами баз Nalicije, Postupil и Vydan. Ниже приведена программа (рис. 3.8), реализующая описанный метод.

Таким образом, на основе команды UPDATE ON можно достаточно эффективно создавать всевозможные учетные системы, где вся сводная информация хранится в главной базе данных, а информация об операциях (поступлении или выдаче) — во вспомогательных базах данных.

```
CLOSE ALL
USE Nalicije
INDEX ON Kod TO Nalicije
USE Postupil
INDEX ON Kod TO Postupil
USE Vydan
INDEX ON Kod TO Vydan
SELECT C
USE Vydan INDEX Vydan
SELECT B
USE Postupil INDEX Postupil
SELECT A
USE Nalicije INDEX Nalicije
UPDATE ON Kod FROM Postupil;
REPLACE Kolicestvo WITH Kolicestvo + B->Postupilo
UPDATE ON Kod FROM Vydan;
REPLACE Kolicestvo WITH Kolicestvo — C->Vydano
```

Рис. 3.8

Кроме операций создания и корректировки информации, для работы с данными всегда необходимы операции удаления. В dBASE с этой целью предусмотрена команда:

DELETE <диапазон> [WHILE <условие>] [FOR <условие>]

С помощью этой команды можно удалить любую запись или диапазон записей, удовлетворяющих заданным условиям. Удаленные записи отмечаются специальным признаком, но по-прежнему остаются в базе.

Команды dBASE LIST или DISPLAY реагируют на удаленные записи в зависимости от системного параметра, устанавливаемого командой:

SET DELETED on/off

Обычно этот параметр находится в положении OFF. В этом случае удаленные записи обрабатываются как и

любые другие, но при выводе отмечаются *. Если установлен режим SET DELETED ON, то большинство команд игнорирует удаленные записи так, как если бы их не было в базе. Если в дальнейшем окажется, что некоторые из записей были удалены ошибочно, то их можно восстановить в базе командой:

RECALL [<диапазон>] [WHILE <условие>]
[FOR <условие>]

Эта команда снимает в записи отметку об удалении. С ее помощью однако, нельзя, восстановить физически удаленные записи. Такое удаление происходит после применения команды PACK.

При этом все записи, отмеченные для удаления, стираются, а занятое ими место освобождается. Индексные файлы, открытые для базы, также будут автоматически откорректированы. Например, для удаления всех записей с кодом R1715 из базы Detali достаточно написать:

DELETE ALL FOR Kod = "R1715"
PACK

В том случае, когда необходимо очистить всю базу, используется быстрая команда ZAP.

3.3.3. Связь между базами

В 2.2.5 мы познакомились с возможностью открывать в разных рабочих областях и обрабатывать одновременно несколько баз данных. Для этого в языке имеются операторы SELECT, позволяющий выбрать рабочую область, и SET RELATION TO, устанавливающий связь между базами. В программе может быть открыто до 10 рабочих областей. Они могут обозначаться либо цифрами от 1 до 10, либо буквами от A до J. Если используется средство Catalog, то рабочая область 10 выделяется для ведения каталога и в распоряжении остается 9 рабочих областей. Выбор текущей рабочей области осуществляется оператором

SELECT <рабочая область/альтернативное имя>

В операнде можно указать либо букву, соответствующую необходимой рабочей области, либо альтернативное имя (алиас) базы данных, открытой в этой области. Например, команда

SELECT 2

выбирает рабочую область 2 или B, что то же самое. При этом база данных, находящаяся в этой области, и все обслуживающие ее файлы (.NDX, .FRM, .FMT и т.д.) становятся текущими. Хотя максимальное количество областей в dBASE III plus равно 10, общее количество одновременно открытых файлов не может превышать числа, заданного в системном наборе операционной системы, называемом Config.sys. Обычно в нем заданы параметры:

FILES = 20
BUFFERS = 15.

Часть буферов и файлов используется операционной системой, поэтому при получении сообщения о невозможности открыть новый файл эти параметры должны быть увеличены.

Для наглядности вместо номеров областей или букв можно использовать имена баз данных, открытых в этих областях, или альтернативные имена этих баз. Например, если в области "B" открыта база Detali с альтернативным именем Basa2:

SELECT B
USE Detali ALIAS Basa2

то в дальнейшем при необходимости сделать область "B" текущей можно писать:

SELECT Basa2

Если альтернативное имя не назначено, то по умолчанию им становится основное имя базы. Например, если написать

SELECT B
USE Detali

то далее в обращениях к области "B" можно использовать альтернативное имя. Например:

SELECT Detali

Иногда имя рабочей области не может быть известно заранее и хранится в виде значения некоторой переменной. Для обращения к такой области можно использовать макрофункцию &. Например, если имя Detali задано в поле Oblast, то команда

SELECT &Oblast

сделает область с базой данных Detali текущей. Выбор текущей рабочей области позволяет работать со всеми полями, принадлежащими находящейся в ней базе данных. Однако часто необходимо обращаться к полям, входящим в записи других баз, содержащихся в других областях. В этом случае перед именем поля ставится имя области и знак "->".

Рассмотрим пример. В табл. 3.8 приведена структура базы Detali. В записях этой базы хранится информация о деталях, произведенных за один день каждым рабочим. Если рабочий в течение дня делал детали двух типов, то в базе будут для него соответственно две записи. Вместо фамилии рабочего используется его учетный номер в картотеке.

Таблица 3.8

Имя поля	Тип	Длина	Дес.поз.
Kod	C	40	
Kolicestvo	N	4	0
Nomer	N	4	0
Data	D	8	

В табл. 3.9 приведена структура базы данных Spravka, в которой находятся описания деталей и расценки.

Таблица 3.9

Имя поля	Тип	Длина	Дес.поз.
Kod	C	40	
Opisan	C	100	
Cena	N	6	2

Существует еще база Kartoteka, представляющая собой картотеку рабочих. Ее структура дана в табл. 3.10

Таблица 3.10

Имя поля	Тип	Длина	Дес.поз.
Nomer	N	4	0
Fam	C	20	
Imia	C	10	
Otcestvo	C	10	

Допустим, что все три базы являются индексированными. Выполнить индексацию можно командами:

```
SELECT A
USE Detali
INDEX ON Kod TO SortDet
INDEX ON Nomer TO Spisok
SELECT B
USE Spravka
INDEX ON Kod TO SortSprv
SELECT C
USE Kartoteka
INDEX ON Nomer TO SortKart
```

Если при работе в области "А" необходимо обратиться в поля базы Kartoteka, то сделать это можно так:

```
SELECT A
DISPLAY ALL Kod,Kolicestvo,C->Fam,C->Imia
```

Действительно, мы получим значение полей Fam и Imia из базы Kartoteka, но оно будет одним и тем же для всех записей базы Detali. Произойдет это потому, что в каждой области dBASE заводит свой собственный указатель записи. Команда DISPLAY воздействует только на указатель области "А", поскольку она используется для вывода всех записей именно этой области. Указатель же области "С" остается неизменным. Поэтому фамилия и имя не меняются. Для правильной обработки нужно установить связь между базами. Это делается оператором:

SET RELATION TO <выражение>
INTO <имя>

Операнд <выражение> может представлять либо значение ключа в альтернативной базе, либо указывать непосредственно номер записи. В последнем случае выражение должно иметь числовое значение. Операнд <имя> задает имя альтернативной области, в которой находится связываемая база. Можно в нем указывать и алиас, если он был задан в команде USE, выданной в этой области.

В 2.2.5. были описаны общие принципы использования этой команды. Здесь же рассмотрим более подробно ее "программистские" возможности. Обратим внимание на способность SET RELATION устанавливать связь по номеру записи. В этом случае не требуется, чтобы альтернативная база была проиндексирована. Поэтому связь может быть установлена с помощью алгоритмов рандомизации, т.е. алгоритмов, вычисляющих номер записи по каким-либо признакам. В простейшем случае, когда две базы должны обрабатываться последовательно, запись за записью, связь может быть установлена с помощью функции RECNO(). Например:

```
SELECT B
USE Basa2
SELECT A
USE Basa1
SET RELATION TO RECNO() INTO B
```

Все операции, выполняемые далее с записями базы Basa1, будут воздействовать на указатель записи в области "В". Но можно применять и другие схемы связи между базами, поскольку оператор SET RELATION TO позволяет использовать любые допустимые числовые выражения. Например, можно, используя связь по номеру записи, организовать структуру данных, называемую "гнездовой список". Такая структура позволяет хранить в базе множественные поля, что не является стандартной возможностью dBASE III plus. На рис. 3.9 представлены структуры баз Golova и Xvost. База Golova содержит записи о заказанном количестве материалов или деталей. Предполагается, что поставки будут осуществляться партиями, ежемесячно. Необходимо хранить дату и величину каждой поставки. Для этого предназначена база Xvost. Названа она так потому, что наименование материалов и заказанное количество хранятся в головной записи в базе Golova. Пользуясь номером этой записи и рандомизационной формулой, можно рассчитать начало цепочки соответствующих записей о поставках ("хвост") в базе Xvost. Любой метод рандомизации всегда содержит алгоритмы инициализации, загрузки и поиска информации. Поэтому на рис. 3.10 представлены четыре подпрограммы: randinit, randnew, randenter и randrep, совместно выполняющие эти функции. Подпрограмма randinit просто очищает базы и подготавливает их к загрузке. Randnew используется для ввода информации о заказе. Каждый раз, когда в базе данных Golova заводится новая (головная) запись, в сопутствующей ей базе Xvost создается пустой "хвост", т.е. цепочка пустых записей. Подпрограмма randenter используется для ввода информации о поступивших материалах. Алгоритм ее работы заключается в поиске свободной записи в "хвосте" и ее заполнении. В качестве примера работы с такой структурой приводится программа randrep, которая выводит отчет о состоянии заказов. В результате ее работы на дисплей выдаются: заказанное количество материалов, полученное количество и их разница.

```
Structure for database: D:Golova.dbf
Number of data recods:   2
Date of last update: 09/19/87
Field  Field Name       Type       Width    Dec
  1    NAIMENOV         Character  40
  2    KOLICESTVO       Numeric     4
** Total **                        45
Record #        NAIMENOV          KOLICESTVO
  1             Деталь 1             1000
  2             Деталь 2              200
```

```
Structure for database: D:Xvost.dbf
Number of data recods: 20
Date of last update: 02/19/87
Field  Field Name     Type     Width  Dec
  1    DATA           Date       8
  2    KOLICESTVO     Numeric    3
** Total **                     12
Record #   DATA        KOLICESTVO
  1        01/07/87        20
  2        02/04/88         7
  3        / /
  4        / /
  5        / /
  6        / /
  7        / /
  8        / /
  9        / /
 10        / /
 11        04/05/88        45
 12        09/04/87       100
 13        / /
 14        / /
 15        / /
 16        / /
 17        / /
 18        / /
 19        / /
 20        / /
```

Рис. 3.9

В 2.2.5. была рассмотрена диалоговая команда **CREATE VIEW**, с помощью которой можно описать связи между существующими базами в диалоговом режиме. В том случае, когда связь устанавливается программным путем, использование диалоговой команды невозможно, поскольку пользователь не имеет представления о том, какие поля и в каких базах эту связь поддерживают.

Для сохранения всей информации о текущем состоянии рабочих областей и связях между ними используется команда:

CREATE VIEW <имя .vue-файла>
FROM ENVIRONMENT

Эта команда создает специальный .vue-файл, включающих информацию о всех рабочих областях, открытых базах данных, индексных файлах, установленных форматах экрана, фильтрах и связях между базами. Чтобы восстановить это состояние в другое время, достаточно выдать команду:

SET VIEW TO <имя .vue-файла>

```
* подпрограмма randinit
CLOSE ALL
USE Golova
ZAP
USE Xvost
ZA
RETURN

* подпрограмма randnew
SET TALK OFF
SELECT A
USE Golova
GOTO bottom
APPEND BLANK
EDIT FIELDS Naimenov,Kolicestvo
SELECT B
USE Xvost
i = 1
DO WHILE i< = 10
  APPEND BLANK
  i = i + 1
ENDDO
RETURN

* подпрограмма randent
SELECT B
USE Xvost
SELECT A
USE Golova

GOTO top
SET RELATION TO (recno()-1)*10 + 1 INTO B
ACCEPT "Введите наименование детали " TO Naimen
LOCATE FOR "&Naimen"$ Naimenov
IF found()
  SELECT B
  STORE recno() TO begin
  DO WHILE Kolicestvo <> 0.AND. recno() < begin + 10
    SKIP
  ENDDO
  IF Kolicestvo <> 0
    ? "Нет места для записи"
  ELSE
    ENDIT NEXT 1
  ENDIF
  SELECT A
ELSE
? "Наименование не найдено"
ENDIF
RETURN

*подпрограмма randrep
SELECT B
USE Xvost
SELECT A
USE Golova
GOTO TOP
SET RELATION TO (recno()-1)*1
DO WHILE .NOT.eof()
  STORE 0 TO Suma
  SELECT B
  SUM NEXT 10 TO Suma
  SELECT A
  ? Naimenov,Kolicestvo,Suma,Kolicestvo-Suma
  SKIP
ENDDO
RETURN
```

Рис. 3.10

Мы рассмотрели принципы и методы использования связей между базами. На самом деле к связанным файлам можно отнести и файлы с memo-полями, поскольку содержимое этих полей хранится в отдельном файле, имеющем расширение .mem. Связь между этим файлом и базой данных поддерживается с помощью поля указателя, имеющего тип memo и хранимого в записях основной базы данных. Длина этого поля всегда 10 байт. "Связанными" можно считать также базу данных и ее индексные файлы. В этом случае каждая запись индексного файла указывает на соответствующую запись в базе данных. Однако, и в первом, и во втором случаях, мы не используем термин "связанные базы", поскольку речь идет об основной базе и ее вспомогательных файлах, а не о двух самостоятельных базах.

3.3.4. Упорядочение и поиск информации

В языке dBASE существуют очень удобные команды для поиска и упорядочения информации в базе: LOCATE, FIND, SEEK, SORT и INDEX. Однако, программируя, нужно учитывать назначение и особенности применения каждой из перечисленных команд. В этом кроются резервы повышения производительности создаваемых систем. В отличие от генераторов программ, программист может учесть специфику каждой отдельной задачи и в зависимости от поставленных целей использовать тот или иной прием программирования. Поскольку поиск информации и поддержание упорядоченности базы являются основными функциями системы обработки данных, любое улучшение характеристик этих функций сразу сказывается на работе всей системы. Попробуем на нескольких примерах разобраться, как лучше применять перечисленные команды в разных ситуациях. Начнем с команды:

LOCATE [<диапазон>] **[WHILE** <условие>]
[FOR <условие>]

Мы уже познакомились с этой командой в 2.3.4 и выяснили, что она действует способом последовательного просмотра всей базы. Поэтому LOCATE не требует упорядоченности базы. Кроме того, у нее есть два мощных операнда WHILE и FOR, которые позволяют очень просто сформулировать условия поиска. Если в области поиска находятся несколько записей, то все они могут быть получены с помощью команды **CONTINUE**.

Операнд FOR команды LOCATE используется для задания условия поиска, а операнд WHILE обычно ограничивает диапазон действия команды. Например, для поиска записи в базе данных Detali, описанной выше, можно выдать команду

```
LOCATE FOR Kod = 25
```

Если в базе находится около 1000 записей, то даже для твердого диска эта операция займет 15-20 с. Это объясняется тем, что команда LOCATE будет последовательно считывать все записи базы в поисках первой записи, удовлетворяющей условию. Для получения остальных записей, удовлетворяющих этому условию, применяется команда CONTINUE:

```
LOCATE FOR Kod = 25
DO WHILE FOUND()
    ? Kod,Kolicestvo,Data
    CONTINUE
ENDDO
```

При этом оператор CONTINUE работает, как и LOCATE, последовательно, но поиск начинается с текущей позиции. Если запись будет найдена, выработается условие "found", которое можно проверить с помощью функции **FOUND()**. Эту функцию удобно использовать в качестве условия выхода из цикла получения всех искомых записей.

Как известно, ускорить процесс поиска можно только с помощью упорядочения информации в базе. В dBASE это можно сделать двумя способами: сортировкой и индексированием. Сортировка базы данных осуществляется командой:

SORT <диапазон> **TO** <файл> **ON** <поле>
[/A] [/E] [/D],...
[WHILE <условие>] **[FOR** <условие>]

Операнд <диапазон> задает диапазон сортируемых записей. Он может принимать значение ALL, NEXT<n>, RECRD<n> или REST.

Операнд <файл> задает имя файла, содержащего отсортированную информацию, <поле> — задает имя поля, по содержимому которого производится сортировка. Ее можно осуществлять в убывающем порядке, если задан параметр /D, или в возрастающем порядке, если задан параметр /A. Параметр /C позволяет не различать прописные и строчные буквы. В команде может быть указано несколько полей сортировки, разделенных запятыми. В этом случае главное поле идет в списке первым. Порядок сортировки для каждого поля может быть индивидуальным. Условия FOR и WHILE позволяют выбрать из базы и отсортировать подмножество записей. Как видим, команда SORT обладает большими возможностями и является очень удобной в применении. Она используется чаще всего для подготовки информации к выдаче на печать или получения сводных отчетов. Например, если нужно отпечатать алфавитный список всех сотрудников учреждения, поступивших на работу в 1987 году, можно подготовить специальный файл Sortbase командами:

```
USE Base
SORT ALL TO Sortbase ON Fio/A;
    FOR Postupil > = ctod("01/01/87")
```

Упорядочение информации в таком отсортированном файле надо поддерживать специальными средствами. Если в файл Spisok будет добавлена новая запись, то порядок сортировки нарушится. Чтобы этого не случилось, можно вставить новую запись в соответствующее ей место командой:

INSERT [BLANK] [BEFORE]

Эта команда вставляет новую запись сразу за текущей (т.е. той, на которую указывает указатель текущей записи). Чтобы новая запись была помещена перед текущей, достаточно применить операнд BEFORE. Команда INSERT работает в режиме полноэкранного редактирования. При этом формат экрана для ввода информации определяется текущим форматным файлом. Очень часто в реальных системах информация перед вводом в базу проходит целый ряд проверок. В этом случае использовать полноэкранный ввод не представляется возможным. Это привело бы к многочисленным ошибкам ввода. Поэтому в команде INSERT предусмотрен операнд BLANK, позволяющий вставлять пустую запись. Затем эта запись может быть заполнена предварительно введенной и проверенной информацией с помощью команды REPLACE. Для поиска места вставки обычно используют команду LOCATE. Например:

```
ACCEPT "Введите фамилию = = >" TO Fam
LOCATE ALL FOR Fam < = Fio
```

```
INSERT BLANK BEFORE
REPLACE Fio WITH Fam
```

Как видим, упорядочение методом сортировки требует специальных приемов поддержания этого состояния и, кроме того, является достаточно длительной процедурой. Поэтому такой метод надо применять в тех случаях, когда сортировку можно произвести заранее, до того момента, когда понадобятся данные из базы. Еще лучше, если на основе отсортированного файла будут выдаваться многочисленные отчеты, обобщающие информацию в базе в разных аспектах. В этом случае получение отчетов может быть выполнено быстрее, чем при применении метода индексных файлов, описанного ниже.

Упорядочение информации с помощью команды SORT не позволяет непосредственно решить проблему ускорения поиска, которая возникла при рассмотрении команды LOCATE. Отсортированные файлы позволяют лишь выводить информацию в определенном порядке, но не способствует более быстрому поиску. Для этого существует другое средство языка — индексные файлы. Они образуются с помощью команды:

INDEX ON <выражение>
    TO <имя файла> [UNIQUE]

Операнды команды INDEX были рассмотрены в 2.2.4, поэтому здесь более подробно остановимся на ее использовании. Эта команда предназначена для упорядочения информации в базе по значению определенного ключа. В этом смысле ее действие подобно команде SORT. Однако в отличие от SORT команда INDEX не переупорядочивает информацию в самой базе. Вместо этого создается новый индексный файл, записи которого состоят из ключей и указателей. Указатели указывают на позицию соответствующей записи в основной базе. Основное преимущество индексного файла заключается в том, что он используется подобно оглавлению книги. Когда необходимо найти запись с определенным ключом в основной базе, у которой есть индексный файл, в dBASE используется команда:

FIND <строка знаков> / <число>

Она осуществляет поиск строки знаков или числа, указанных в операнде, среди ключей записей в активном индексном файле. Это происходит очень быстро файле, поскольку индексный файл всегда является упорядоченным (этим занимается сама система). Кроме того, так как записи индексного файла короткие, система пытается расположить большинство из них в имеющейся свободной оперативной памяти. Поэтому операции с индексом часто вообще не требуют обращения к внешним устройствам. После быстрого поиска нужного ключа в индексном файле из его записи извлекается указатель на соответствующую запись в базе данных и прямым обращением к базе эта запись помещается в оперативную память. Таким образом, если для поиска в файле, имеющем 1000 записей, применить команду LOCATE — FOR, то понадобится в среднем 500 считываний с диска. Если же при поиске использовать индексный файл, понадобится некоторое небольшое время для поиска в индексе и всего одно считывание с диска.

Операнд команды FIND представляет собой строку знаков. В отличие от строковых констант она не обязательно должна заключаться в апострофы. По этой причине в операнде команды нельзя использовать переменную. Указанная строка сравнивается с ключом записи. Если необходимо точное совпадение, нужно устанавливать режим

SET EXACT ON

Если в начале ключа есть разделители (например, пробелы), достаточно заключить операнд команды в кавычки. В этом случае при сравнении будут учитываться все разделители, находящиеся в операнде.

В операнде можно использовать символьную переменную. Для этого перед ней нужно поставить знак макрофункции. Например:

```
STORE "ключ" TO Keyvalue
FIND &Keyvalue
```

Эта последовательность команд аналогична написанию

```
FIND ключ
```

Можно в операнде FIND использовать и цифровой ключ. Он воспринимается как строка знаков, изображающая число. При этом поиск в индексном файле для полей типа numeric выполняется правильно. Чтобы в этом случае использовать переменную, нужно преобразовать ее в символьный вид. Например:

```
STORE 23 TO Key
STORE STR(Key) TO Keyvalue
FIND &Keyvalue
```

Использовать непосредственно команду

```
FIND &Key
```

нельзя, так как макрофункция не применяется к числовым переменным.

Недостатком применения индексных файлов является увеличение объема необходимой оперативной памяти, поскольку наряду с записями из баз данных нужно хранить еще и индексы. Но это не самое серьезное ограничение. Значительно хуже то, что в отличие от LOCATE, в которой можно задавать любое условие поиска в операнде FOR, при использовании индексного файла можно осуществлять поиск только по тем полям базы, для которых созданы индексные файлы. Поэтому, если нужно составлять различные запросы к базе, включающие поиск по разным полям, приходится создавать несколько индексных файлов. Количество индексных файлов к одной базе данных не ограничивается. Однако одновременно открытыми (используемыми) их может быть не более 7 для одной базы. Рассмотрим этот вопрос подробнее. Если необходимо осуществить поиск в базе по двум разным полям, придется завести два индексных файла. Например, для поиска в базе Detali (см.3.3.3) записей по коду детали можно создать индексный файл Pokodu с помощью следующих команд:

```
USE Detali
INDEX ON Kod TO Pocodu
```

а для поиска записей по учетным карточкам рабочих придется создать индексный файл Ponomeru:

INDEX ON Nomer TO Ponomeru

Для поиска по этим полям основную базу нужно открыть с одним из индексов. Например:

USE Detali INDEX Pokodu

или

USE Detali INDEX Ponomeru

Можно открыть основную базу сразу с несколькими индексными файлами, например:

USE Detali INDEX Pocodu,Ponomeru

Но поиск в базе можно производить только по полю первого индекса. Чтобы не приходилось открывать и закрывать файл базы данных, для смены индекса в таких случаях используется команда:

SET ORDER TO <выражение>

где <выражение> соответствует номеру открытого индексного файла, по которому будет вестись поиск или выборка. Если значением выражения будет 0, то поиск будет производиться без индексных файлов. После выдачи команды SET ORDER для активизации индексного файла достаточно выдать любую команду позиционирования указателя текущей записи (GOTO, FIND или SEEK):

```
USE Detali INDEX Pocodu,Ponomeru
ACCEPT "Введите учетный номер" TO Key
SET ORDER TO 2
FIND &Key
```

Как уже говорилось, количество индексных файлов, созданных для одной базы, не ограничивается, хотя одновременно открытых файлов должно быть не более 7. Возникает вопрос: зачем открывать одновременно несколько индексных файлов, если для поиска все равно используется только один из них? Ответ заключается в том, что для автоматического соблюдения упорядоченности индексного файла его нужно держать открытым. Если в базе данных происходят изменения, т.е. в нее добавляются или исключаются записи, то это должно отражаться и в связанном с ней индексном файле. В противном случае информация в индексе не будет соответствовать информации в базе. Именно поэтому dBASE требует, чтобы в момент добавления или удаления записей, либо при изменении значения ключевого поля связанные с базой индексные файлы были открыты. Из этого следует, что программисту приходится каждый раз выбирать между удлинением времени внесения изменений (из-за нескольких одновременно открытых индексных файлов) и затратой дополнительного времени на переиндексацию неоткрытых индексных файлов (после изменений в базе). Чтобы привести в соответствие информацию в индексном файле и основной базе, в dBASE используется команда:

REINDEX

Эта команда переупорядочивает все открытые индексные файлы, связанные с активной базой данных. Например, если во время изменений в базе Detail файл Ponomeru не был открыт, то привести его в соответствие новому состоянию базы можно командами:

```
USE Detali
SET INDEX TO Ponomeru
REINDEX
```

Поскольку операция переиндексации занимает довольно много времени (сравнима с SORT), то перед выдачей команды REINDEX надо позаботиться о том, чтобы все ненужные индексные файлы были закрыты. Сделать это можно командами:

CLOSE INDEX

или

SET INDEX TO <Enter>

Уже упоминалось о том, что для поиска в базе с использованием индекса можно применить только один ключ. Но часто бывает так, что в запросе содержится не одно, а несколько полей. В этом случае, если комбинация полей не меняется, можно применить составной ключ. Например, если в базе есть поле Familija и поле Imia, можно выполнять поиск по фамилии и имени одновременно, если индексирование выполнить командой:

INDEX ON Familija + Imia TO Famim

Далее для поиска записи можно использовать составной ключ:

```
USE Kartrab INDEX Famim
Accept "Введите фамилию = = >" TO Fam
Accept "Введите имя = = >" TO Im
STORE Fam + Im TO Key
FIND &Key
```

Однако, таким способом не всегда можно решить проблему. Например, когда нужно найти запись, у которой поле Kod попадает в некоторый интервал. В этом случае тоже существуют два поля, влияющие на поиск записи, но составной ключ применить не удастся. Запрос на поиск должен формироваться иначе. Первую запись, удовлетворяющую этому условию, найдем с помощью команды:

SEEK <выражение>

Эта команда находит первую запись, чей ключ удовлетворяет заданному выражению. Если такой записи в базе нет, указатель устанавливается на конец файла (значение функции EOF() = .T., а функция FOUND() = .F.). Как видим, SEEK отличается от FIND тем, что позволяет вместо жесткого шаблона задавать выражение, значение которого будет сравниваться с ключом. Сравнение всегда выполняется слева направо, начиная с первого символа в ключе. Если требуется полное совпадение, то тут, как и в команде FIND, нужно выдавать

SET EXACT ON

Использование выражений облегчает программирование и упрощает программы. Например, используя SEEK для поиска в цифровом индексе, достаточно написать

SEEK Key

Команды SEEK и FIND могут использоваться с операндом WHILE команд последовательного поиска или просмотра (EDIT, DISPLAY, LOCATE и т.п.) для организации быстрой выборки всех записей, удовлетворяющих сложным условиям. Этот метод аналогичен по своей функции операнду FOR последовательных команд, но дает огромный выигрыш в скорости выборки.

Идея заключается в том, чтобы выделить основное множество записей, подлежащих обработке, используя для этого индексный файл. После того, как первая запись из этого множества будет найдена, можно, применяя операнд WHILE команд последовательного поиска и просмотра, отобрать все записи, удовлетворяющие любому возможному условию. Так как при этом записи базы будут просматриваться в соответствии с индексным файлом, т.е. в порядке возрастания ключей, никаких лишних считываний записей из базы данных не будет. Поясним это примером.

Допустим, нам нужно отобрать и показать все записи, попадающие в указанный диапазон ключей. С этой целью база данных должна быть упорядочена по возрастанию кода записи:

```
USE Detali INDEX Kod && Поле индекса символьное
ACCEPT "Введите нижнюю границу = >" TO Low
ACCEPT "Введите верхнюю границу = >" TO High
SEEK Kod > = Low
IF FIND()
    LIST WHILE Kod < = High
ELSE
    ? "Записей в указанных пределах нет"
ENDIF
```

Как видим, в программе для вывода найденных записей используется команда LIST. Однако, поскольку она работает с индексированным файлом, записи будут выбираться с помощью индекса. Установку на первую запись выполнит команда SEEK, а контролировать отбор только нужных записей будет операнд WHILE. Такой прием работы с упорядоченными базами данных позволяет выбрать все необходимые записи даже в том случае, когда в базе имеется несколько записей с одинаковыми ключами. Операнд WHILE можно успешно применять в операторах: EDIT, COUNT, SUM, REPORT, COPY и т.п. При этом нужно использовать тот же метод для ускорения работы: первую запись в базе находить с помощью FIND или SEEK, а остальные обрабатывать с помощью операнда WHILE соответствующей команды.

Приведем еще один пример. Нужно подсчитать количество деталей, сделанных одним рабочим, основываясь на информации, хранимой в базе Detali. Применим для этого оператор SUM. Программу составим с использованием изложенного метода:

```
USE Detali INDEX Ponomeru && Поле индекса цифровое
ACCEPT "Введите учетный номер рабочего = >" TO Un
FIND &Un
SUM Kolicestvo TO Suma WHILE Nomer = VAL(Un)
? "Всего сделано",Suma,"деталей"
```

Если предположить, что в базе данных Detali находится 1000 записей, то приведенная программа по сравнению с оператором

```
SUM Kolicestvo TO Suma FOR Nomer = VAL(Un)
```

даст выигрыш во столько раз, во сколько общее число записей превышает количество записей с полем Nomer, равным значению Un.

Если условия отбора записей известны заранее, то можно воспользоваться стандартным средством фильтрации записей. Это делается с помощью команды:

SET FILTER TO <условие>

Команда служит хорошим дополнением к возможности поиска по индексу или отбору записей с помощью операнда WHILE. Например, если нужно отобразить все записи о деталях определенного рабочего за известный отрезок времени, придется пользоваться индексным файлом Ponomeru. Это позволит быстро найти и выбрать все записи одного рабочего. Чтобы из них отобрать только те, которые попадают в определенный орезок времени, можно воспользоваться фильтрацией:

```
USE Detali INDEX Ponomeru
STORE 0 TO Un
INPUT "Введите учетный номер" TO Un
ACCEPT "Введите начало периода" TO Dn
ACCEPT "Введите конец периода" TO Dk
SET FILTER TO Data < = ;
      CTOD(Dk) .AND. Data > = CTOD(Dn)
SEEK Un
LIST ALL Kod, Kolicestvo WHILE Nomer = Un
```

В заключении приведем пример реализации подпрограммы поиска записей в картотеке. В ней сочетаются различные типы команд поиска и отображения информации. Для отбора карточки по нескольким критериям используется метод синтеза программы. Оператор EDIT и его операнды создаются на основе полученной от пользователя информации. При поиске карточки по фамилии автора пограмма использует индексный файл, если он открыт, либо осуществляет последовательный поиск, если база данных не упорядочена. Для быстрого просмотра используется стандартное средство системы — команда BROWSE.

```
* Модуль KartoPO (поиска/просмотра карточек)
* Выполняет поиск как по фамилии,
* так и по содержимому полей карточки
IF "<dbf()
DO WHILE .T.
 SET BELL OFF
 SET FORMAT TO
 STORE 4 TO Vybor
 CLEAR
 @ 2, 0 TO 18,79 DOUBLE
 @ 3,28 SAY [Как будете искать карточки?]
 @ 4,1 TO 4,78 DOUBLE
 @ 7,25 SAY [1. По фамилии автора]
 @ 9,25 SAY [2. По реквизитам]
 @ 11,25 SAY [3. Просматривать]
 @ 13,25 SAY [4. Вернетесь в основное меню]
 @ 17,25 SAY "Введите номер"
 @ 17,40 GET Vybor PICTURE "9" RANGE 1,4
 READ
 DO CASE
  CASE Vybor = 4
   EXIT
  CASE Vybor = 1 && Поиск по фамилии
   CLEAR
   STORE SPACE(40) TO Fio
   @ 10,5 SAY [Введите фамилию автора] GET Fio
```

```
   READ
   SET FORMAT TO Kartosay
   IF " < NDX(1)   && Если есть индексный файл
    SET EXACT OFF
    SEEK TRIM(LTRIM(Fio))
    IF found()
     EDIT ALL FOR TRIM(LTRIM(Fio))$AU
   CLEAR
   @ 10,1 SAY " Больше авторов с такой фамилией нет "
   WAIT [Для возврата в меню нажмите любую клавишу]
  ELSE
   @ 10,1 SAY " Автор в картотеке не найден. "
   WAIT [ Для возврата в меню нажмите любую клавишу]
  ENDIF
  SET INDEX TO
 ELSE
  LOCATE ALL FOR TRIM(LTRIM(Fio))$AU
  IF found()
   DO WHILE found()
    EDIT NEXT 1
    CONTINUE
   ENDDO
  ELSE
   @ 10,1 SAY " Автор в картотеке не найден. "
  WAIT [   Для возврата в меню нажмите любую клавишу]
  ENDIF
ENDIF && если есть база
CASE Vybor = 2 && Поиск по реквизитам
 STORE SPACE(20) TO GSI,GAU,GZA,GPZ,GST,GIZ,GOI
 STORE 0 TO Gib,Gie
 SET CONFIRM ON
 SET FORMAT TO kartoask
 READ
 SET CONFIRM OFF
 cmd = "EDIT ALL FOR .T."
 IF LEN(TRIM(Gsi)) < > 0
  cmd = cmd + ".AND.TRIM(Gsi)$Si"
 ENDIF
 IF LEN(TRIM(Gau)) < > 0
  cmd = cmd + ".AND.TRIM(Gau)$Au"
 ENDIF
 IF LEN(TRIM(Gza)) < > 0
  cmd = cmd + ".AND.TRIM(Gza)$Za"
 ENDIF
 IF LEN(TRIM(Gpz)) < > 0
  cmd = cmd + ".AND.TRIM(Gpz)$Pz"
 ENDIF
 IF LEN(TRIM(Gst)) < > 0
  cmd = cmd + ".AND.TRIM(Gst)$St"
 ENDIF
 IF LEN(TRIM(Giz)) < > 0
  cmd = cmd + ".AND.TRIM(Giz)$Iz"
 ENDIF
 IF LEN(TRIM(Goi)) < > 0
  cmd = cmd + ".AND.TRIM(Goi)$Oi"
 ENDIF
 IF Gib < > 0.OR Gie < >
  cmd = cmd + ".AND.Gib < = Gi.AND.Gi < = Gie"
 ENDIF
 SET FORMAT TO kartosay
 &cmd
 SET FORMAT TO
 CLEAR
@ 10,5 SAY "Больше записей с указанными признаками нет"
 @ 21,5
 WAIT "Для возврата в меню нажмите любую клавишу"
CASE Vybor = 3 && Просмотр
 @ 20,1 clear
@ 20,10 SAY "Просмотр в формате карточек? (Y/N) = = >"
 WAIT "" TO Answer
 IF Answer $ "YyДд"
  SET FORMAT TO Kartosay
  EDIT ALL
  SET FORMAT TO
 ELSE
  SET MENUS OFF
  GOTO top
  BROWSE WIDTH 30 NOAPPEND FIELDS si,au,za,gi
 ENDIF
ENDCASE
ENDDO T
ELSE
@ 20,5 SAY "Сначала выберите, пожалуйста, картотеку"
ENDIF
RETURN
```

АРМ для инженеров-машиностроителей

Возможности современных технических и программных средств персональных ЭВМ IBM PC XT, AT и совместимых с ними позволяют эффективно автоматизировать конструкторскую и технологическую подготовку производства.

В настоящее время в качестве фактического стандарта на программное обеспечение для выпуска конструкторских чертежей на ПЭВМ утвердилась система AutoCAD.

Она поддерживает процесс черчения изделий на экране ПЭВМ, ведение графической базы данных и вывод на стандартные принтеры и графопостроители.

При этом обеспечивается некоторое повышение производительности черчения и значительное повышение качества графики. Настройка пакета AutoCAD на специфику стандартов ЕСКД еще более повышает производительность черчения. Вводя в ПЭВМ параметрические программы черчения типовых элементов чертежей, можно обеспечить дальнейшее повышение удобства работы конструктора. Эти задачи решены в комплексном программном средстве "АРМ конструктора".

АРМ конструктора представляет собой надстройку к известному графическому пакету AutoCAD v. 10.0 English Language.

АРМ конструктора обеспечивает:

- выбор из меню и автоматизированное черчение форматов ЕСКД А1, А2, А3, А4 с низким и высоким штампом;
- автоматизированное черчение обозначений: шероховатости обработки, покрытия швов сварных соединений, паянных и клеенных соединений, стандартных клепаных соединений;
- автоматизированное черчение обозначений допусков: прямолинейности, плоскостности, круглости, цилиндричности, профиля продольного сечения, параллельности, перпендикулярности, наклона, соосности, симметричности, позиционного пересечения осей, радиального биения, торцевого биения, биения в заданном направлении, полного торцевого биения, формы профиля, формы поверхности;
- автоматизированное черчение типовых отверстий на чертежах: отверстий с фаской, отверстий с двумя фасками, отверстий с резьбой, отверстий с цековкой, отверстий с цековкой и фаской, отверстий с цековкой и резьбой, отверстий с конусом;
- стилизованные шрифты для выпуска плакатов;
- программы простановки размеров в соответствии с ЕСКД: горизонтальных линейных размеров, вертикальных линейных размеров, радиусов, диаметров;
- полный набор программ для размещения и редактирования текста на поле чертежа (выпуск таблиц и технических требований), куда входят: программа загрузки текста из произвольного текстового файла в чертеж, программа выгрузки текста из чертежа в произвольный текстовый файл, программа увеличения текста для просмотра (текстовая линза), программа редактирования текста на чертеже стандартными текстовыми редакторами, программа, облегчающая выпуск текстовых плакатов, программа разбивки текстовых строк на отдельные слова в поле чертежа;
- автоматизированное черчение крепежных деталей в широкой номенклатуре: болтов с шестигранной головкой, болтов с шестигранной уменьшенной головкой, болтов с шестигранной уменьшенной головкой, с полем допуска h9 и короткой нарезной частью, болтов с шестигранной головкой для шарнирных соединений, болтов со шлицем в шестигранной головке, невыпадающих болтов со шлицем в шестигранной головке, невыпадающих винтов с полукруглой головкой, винтов с цилиндрической головкой, винтов с контровочным отверстием в цилиндрической головке, винтов с потайной головкой, полем допуска h9 и короткой нарезной частью, винтов с потайной головкой, винтов с потайной головкой и крестообразным шлицем, винтов с полукруглой головкой, винтов с с полукруглой головкой, с полем допуска h9 и короткой нарезной частью, винтов с плосковыпуклой головкой, винтов с плосковыпуклой головкой и крестообразным шлицем, винтов для люков, винтов с полупотайной головкой (угол 90 градусов), винтов с полупотайной головкой (угол 120 градусов), винтов с цилиндри-

ческой головкой и шестигранным углублением под ключ, винтов установочных с коническим концом, винтов установочных с плоским концом, винтов установочных с цилиндрическим концом, шпилек, гаек шестигранных, гаек шестигранных низких, гаек шестигранных прорезных, гаек шестигранных прорезных низких, штифтов цилиндрических, штифтов конических, шайб пружинных, шайб.

Система позволяет автоматически чертить не только стандартные, но и оригинальные крепежные детали. Особенно полезной библиотека параметризованного крепежа, входящая в предлагаемый АРМ, оказывается при выполнении перечерчивания. Имеются справочные таблицы прочностных характеристик крепежных деталей. Для различных материалов здесь приведены: таблицы допустимых нагрузок на разрыв резьбы, таблицы допустимых усилий и моментов затяжки резьбы, таблицы допустимых нагрузок на срез стержня.

Система предназначена для применения в конструкторских подразделениях при автоматизированном выпуске чертежной КД с использованием графического пакета AutoCAD.

Предлагаемая программная система, написанная на языке AutoCAD, поставляется в виде исходных текстов и документации. Исходные тексты меню и программ снабжены комментариями на русском языке и могут служить иллюстрацией методов разработки прикладных систем в среде ППП AutoCAD.

Все программы обильно иллюстрированы справочными слайдами, обращение к программам осуществляется через меню.

Использование данного АРМ существенно повышает производительность труда конструктора при автоматизированном конструировании с использованием пакета AutoCAD, высвобождает его время для решения творческих задач.

Функциональная схема АРМ конструктора представлена на рис.1.

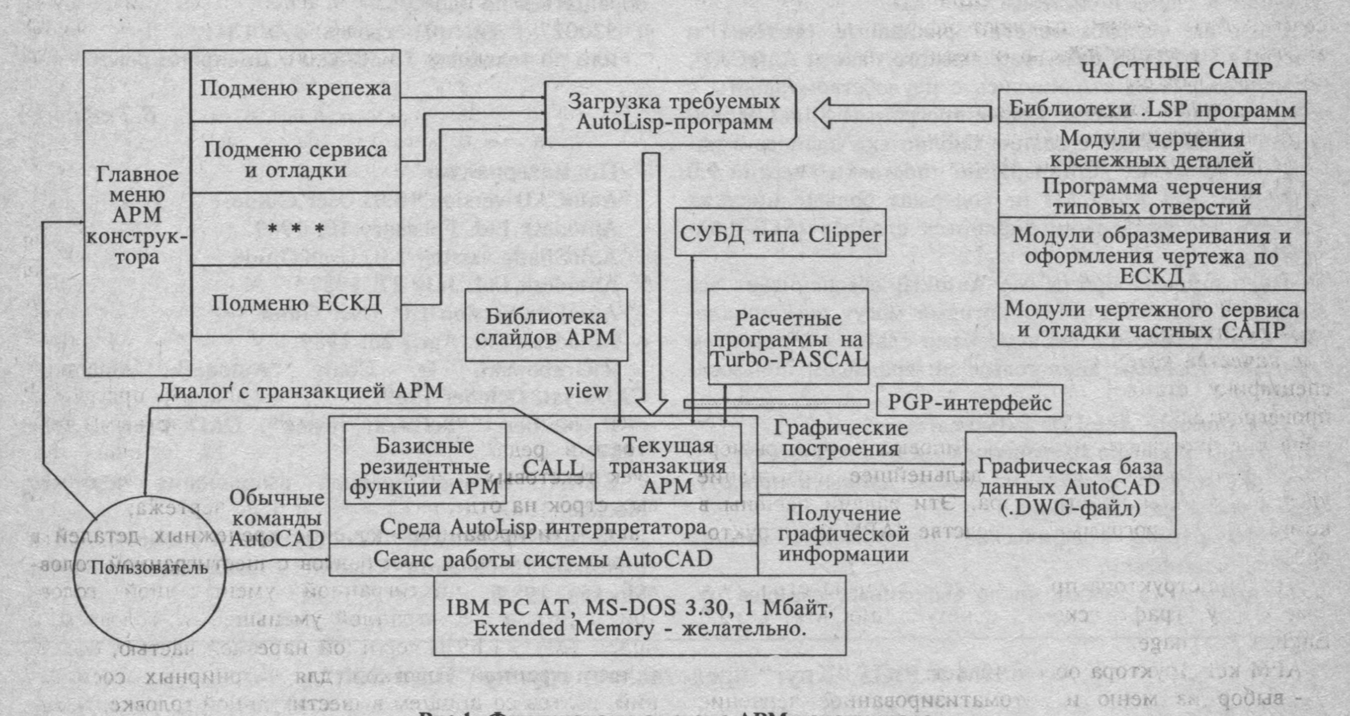


Рис.1. Функциональная схема АРМ конструктора.

Параметрические программы избавляют конструктора от необходимости черчения деталей каждый раз при внесении конструктивных изменений. Задание чертежа в виде параметрической программы сводит процесс внесения изменений к вызову этой программы и подстановке параметров в диалоге.

Приведем краткую характеристику опробованных нами программных средств САПР для ПЭВМ. Общая черта описанных ниже программ состоит в том, что все они совместимы по форматам файлов с AutoCAD и предназначены для совместной с ним эксплуатации.

AutoShade — программа построения трехмерных цветных видов по трехмерным чертежам AutoCAD. Качество полученных изображений сильно зависит от цветового диапазона графического адаптера (рекомендуется применение адаптера VGA). Входные данные должны быть представлены в виде файлов FLM, формируемых AutoCAD. Каждый FLM-файл состоит из отдельных сцен. Сцена характеризуется изображением и расположением источников света. Закраска сцен производится последовательно под управлением пользователя.

AutoFlix — программа построения рекламных компьютерных фильмов по выходным материалам AutoCAD и AutoShade.

Abase — база данных для AutoCAD.

Glisp — генератор параметрических САПР по чертежам AutoCAD.

AutoManager — программа управления машинной базой чертежей и просмотр чертежей отдельно от AutoCAD.

AutoSave — программа, обеспечивающая совместимость версий AutoCAD и восстановление после сбоев.

AutoSolid — система пространственного моделирования трехмерных деталей и сборочных единиц.

Допускает подключение пакетов конечно-элементного моделирования.

Autodesk Animator — совместимый с AutoCAD пакет редактирования видеоизображений, полученных с фиксатора кадров.

Autosketch — простой быстродействующий графический редактор предназначен для создания эскизов.

Нами разработан программный продукт AutoLIB.

AutoLIB — новейшее программное средство, совместимое с AutoCAD, обеспечивает просмотр и ведение библиотек слайдов системы AutoCAD.

Если вы разрабатываете прикладные системы и частные САПР на базе графического пакета AutoCAD, то наверное уже столкнулись с неудобством работы с библиотеками слайдов. Кроме программы SlideLIB, которая обеспечивает создание библиотеки слайдов в пакетном режиме, стандартные поставки версий 9.0 и 10.0 пакета AutoCAD не содержат больше никаких средств для поддержки библиотек слайдов (SLB-файлов).

Предлагаемая программа AutoLIB обеспечивает все функции и возможности, которые могут вам понадобиться при работе с библиотеками слайдов и при этом обладает удобным диалоговым интерфейсом пользователя.

При помощи AutoLIB вы можете:

- открыть любую имеющуюся -SLB- библиотеку;
- просмотреть оглавление любой длины с прокруткой в обоих направлениях;
- просмотреть графическое изображение любого слайда библиотеки;
- уничтожить любое число выбранных слайдов путем расстановки меток;
- вывести слайд в отдельный файл без удаления его из библиотеки;
- добавить один или несколько слайдов в библиотеку извне;
- просмотреть список слайдовых файлов или библиотек слайдов любого каталога, не выходя из AutoLIB и текущей библиотеки слайдов;
- просмотреть графическое изображение любого внешнего слайда, не прерывая работы с текущей библиотекой.

Все операции обновления библиотеки выполняются быстро, независимо от ее объема, так как AutoLIB обращается к диску на самом нижнем уровне.

Справочная информация (HELP) и все сообщения AutoLIB выдает на русском и английском языках.

Программные продукты “АРМ конструктора-машиностроителя“ и AutoLIB v. 1.0 коммерчески доступны.

По вопросу приобретения этих продуктов следует обращаться по адресу:

320027, г.Днепропетровск, а/я 1341
или по телефону 58-58-32 в г.Днепропетровске.

Б.Ткаченко

По материалам:

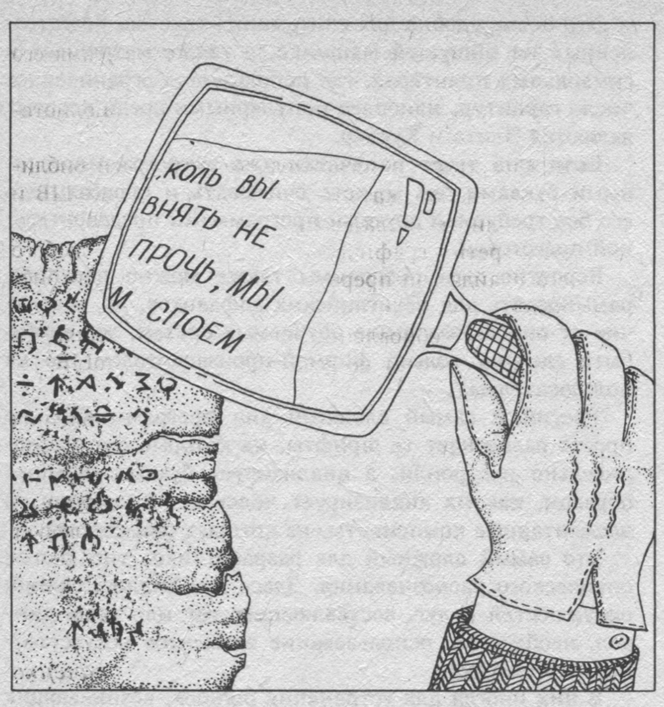
AutoCAD version 10.0. User Guide.
Autodesk Ltd. February 10, 1989.
AutoShade version 1.0. User Guide.
Autodesk Ltd. July 27, 1987.
AutoFlix version 1.0. User Guide.
Autodesk Ltd. April 23, 1989.

R.Grabowski, D. Cohh “Autodesk Animator“, CADalyst, October 1989.

P. Metherel “Softwear review“, CAD User, October 1989.

ЕСКД. Общие правила выполнения чертежей. Москва, 1988.

Детали крепежные. Конструкция и размеры. Технические требования. Сборник отраслевых стандартов. Москва, 1985.



Джон Маккормик, шеф Вашингтонского бюро Newsbytes News Network любезно предоставил редакции КомпьютерПресс материалы, основанные на собственных впечатлениях, о работе с системами оптического распознавания символов.

Оптическое распознавание символов

Мое близкое знакомство с областью оптического распознавания символов (ОРС) состоялось в 1987 году, когда я на основе специально для этой цели разработанных тестов написал для журнала Byte (апрель 1987 г.) статью-обзор всех имевшихся в то время систем ОРС.

Тогда ни одна из испытанных систем не могла сканировать тексты типографских изданий (книг, газет и журналов), хотя две из них — PCS фирмы CompuScan и OCR scanner компании Intelligent Optics — прекрасно обрабатывали машинописный текст, не содержащий графики.

Обе системы до сих пор продаются и по-прежнему хорошо работают с текстами, напечатанными на непереплетенных страницах, но за последние несколько лет системы оптического распознавания символов чрезвычайно сильно улучшили свои параметры.

До недавнего времени единственными хорошо работающими системами ОРС на микрокомпьютерах были лишь те, которые распознавали напечатанный в одну колонку на машинке или ромашковом принтере текст без графических добавлений к нему.

За бортом оставались многие интересные книжные и журнальные материалы. Их было практически невозможно перевести в компьютерную форму, так как они все были набраны в типографиях, а в журналах, кроме того, сверстанный в несколько колонок текст сочетался с большим числом картинок.

Благодаря последним успехам в программировании и, особенно, значительному повышению быстродействия микрокомпьютеров на рынке появились относительно недорогие программы, которые могут обрабатывать даже журнальные страницы. Газеты сканировать по-прежнему тяжело — буквы часто просвечиваются через тонкую бумагу на другую сторону.

Преобразование информации в текстовую форму делает имеющуюся в документе информацию более пригодной для дальнейшей обработки, поскольку в этом случае возможно использование всех многочисленных средств обработки текстов — редакторов, баз данных и т.п.

Конечно, с технической точки зрения проще хранить на диске оцифрованную копию документа, но при этом информация занимает гораздо больше места, и работа с ней затруднена.

Даже если сканированные изображения для облегчения поиска собраны в базу данных и снабжены ключевыми словами, все равно это не все слова, а только те, которые человек, вводивший документ, считал важными.

Обработанные же программой ОРС материалы хранятся в виде их полных текстов и найти любую содержащуюся в них информацию не составит труда.

Типы программ ОРС

Системы ОРС могут состоять из сканера и программы, либо включать лишь программную часть.

Активно разрабатываемые сейчас программы распознавания просто берут графическое изображение, полученное одним из многочисленных сканеров, и преобразуют его в текстовый файл.

Более старые системы включали в себя сканер и специальную программу, которая могла работать лишь с одним этим типом сканера. Сам процесс преобразования изображений в символы производился на специальном микрокомпьютере — либо встроенном внутри сканера, либо расположенном на интерфейсной плате, подключаемой между сканером и компьютером.

Эти сложности вызывались тем, что мощности компьютеров просто не хватало для выполнения самых элементарных операций за приемлемое время, и приходилось ставить отдельный быстрый микропроцессор для обработки изображения.

К этому типу относится очень надежный сканер PCS, продукт компании CompuScan. Он может обработать стопку из 50 машинописных (не типографских) страниц без участия оператора.

Более новый тип систем распознавания — чисто программный. Они могут читать файлы, введенные большим числом сканеров и даже принятые по факсу, хотя низкое качество факс-копий и их низкая разрешающая способность обычно выливаются в плохие результаты преобразования.

Существует три основных типа ОРС-программ. Первый тип, самый простой в разработке и отнимающий массу времени при использовании — это программируемые (обучаемые) системы. Перед тем как вы сможете начать использовать такую программу для дела, вам придется потратить от 30 минут до нескольких часов, обучая ее всем типам шрифтов, которые вы собираетесь считывать.

Преимуществом этого типа систем является то, что они могут читать практически любые тексты, состоящие из букв любых алфавитов, и при работе с одинаково отпечатанными документами их можно даже предпочесть более сложным и дорогим системам.

Но из-за того, что в книгах, газетах и журналах используется достаточно большое число различных гарнитур (видов шрифта), а программы этого типа могут распознавать лишь тот тип букв, которому они были обучены, вам придется повторять процесс обучения для каждого вновь встречающегося типа шрифта.

Но если вам надо вводить в компьютер тексты, напечатанные всего одним-двумя шрифтами, то может оказаться, что обучаемые системы ОРС дают меньше ошибок, чем содержащие элементы искусственного интеллекта и распознающие многочисленные шрифты.

Хотя лично я никогда не пробовал, обучаемые программы можно настроить и на распознавание русских шрифтов.

Следующий тип систем использует жестко запрограммированные наборы символов (обычно 10-12 шрифтов). Это очень удобно для считывания текстов, подготовленных на пишущей машинке, а также матричных и символьных принтерах, где используется ограниченное число гарнитур, наиболее популярными среди которых являются Элита и Курьер.

Если ваш текст напечатан уже запрограммированными буквами, вы можете считывать и обрабатывать его без требуемой другими программами предварительной подготовки.

Вероятно, эти программы также можно перепрограммировать для неанглийских алфавитов, но, в отличие от описанных выше обучаемых систем, это может быть сделано только фирмой-производителем, но не пользователем.

Третий и самый сложный вид программ ОРС не просто распознает те шрифты, на которые его предварительно настроили, а анализирует буквы таким же образом, как их анализирует человек — разбивая на элементарные компоненты, из которых они состоят.

Это самый сложный для разработчиков тип систем оптического распознавания. Здесь для анализа линий, окружностей и дуг, составляющих тот или иной символ, необходимо использование элементов искусственного интеллекта.

В них иногда для устранения ошибок, возникающих при неправильном прочтении слов, используют даже встроенные словари.

Если вы сканируете английский текст, программа отметит слово, которое она не смогла разобрать. Она исходит из того, что исходный текст не содержит ошибок в правописании, и все ошибки сделаны ей самой в процессе распознавания символов.

Медики, бизнесмены, адвокаты и ученые могут использовать и собственные дополнительные словари для повышения точности компьютерного чтения.

Этот тип программ, называемый "омнифонт" (распознающий все шрифты), является самым автоматизированным и самым универсальным типом ОРС для латинского алфавита и может преобразовывать практически любые графические материалы. Но несмотря на это, документы, выполненные только одним шрифтом, будут по-прежнему читаться лучше и быстрее, чем другие.

Развитие этих программ до того уровня, на котором они существуют сейчас для английского алфавита, заняло несколько лет, и я подозреваю, что перенос их на другие алфавиты является достаточно сложным несмотря даже на то, что основы используемой технологии остаются неизменными. Кроме того, эти программы требуют для работы больших ресурсов компьютера — распознавание многочисленных шрифтов требует большого объема вычислений.

Самые совершенные программы могут даже автоматически отделять текст от графики в сканируемых страницах и без участия оператора преобразовывать в текстовый файл материалы, напечатанные на странице в несколько колонок.

Сейчас самые распространенные системы ОРС работают на быстрых компьютерах Mackintosh и PC с

микропроцессором 386 или 486, либо поставляются вместе со специальными платами-ускорителями, обычно содержащими микропроцессор MC680xx фирмы Motorola — тот, который используется в Mackintosh.

К сожалению, во время проведения независимых испытаний различных систем ОРС я обнаружил, что заявления производителей о точности и скорости работы их программ часто очень сильно отличаются от действительных результатов. Хотя некоторые фирмы и утверждают, что их продукт делает лишь "две ошибки на 100000 символов", но даже лучшая из испытанных мною систем делала от трех до 20 ошибок на странице.

Если текст отпечатан с невысоким качеством, ошибок обычно бывает еще больше, хотя в зависимости от типа обрабатываемого документа не все эти ошибки будут критичными.

Проблемы создания систем ОРС

Чтобы понять, насколько технически тяжело производить оптическое распознавание символов, попробуем бросить беглый взгляд на технологию печати. В этой статье я буду приводить примеры из английского алфавита — того, на котором я тестировал большинство программ распознавания символов.

Глаз человека может легко различать сотни и тысячи типографских стилей. Читая этот журнал, вы видите, что даже статьи набраны различными шрифтами, не говоря уже о рекламных объявлениях.

Различные пишущие машинки и компьютерные принтеры используют различные "наборы символов" или "гарнитуры", состоящие из всех используемых букв, цифр и символов.

Разные буквы в каждой гарнитуре имеют разную форму, хотя некоторые из них могут быть похожи друг на друга.

На машинописных страницах все буквы обычно имеют одинаковый размер и форму, стандартно и расстояние между ними. Все это делает такие тексты очень простыми для преобразования в компьютерный вид системой ОРС, хотя все равно остаются проблемы с буквой О и цифрой 0, большими и маленькими буквами "M" и "m", "N" и "n".

Если же правый и левый края текста выровнены (выключка текста), тогда расстояния между словами и даже отдельными буквами будут меняться за счет дополнительных пробелов между словами, либо микроподстройки положения каждой буквы в слове.

Так выглядит типичный типографский текст. Он обычно очень сложен для простых ОРС-программ, поскольку затруднения возникают уже при определении начала и конца слов и отдельных букв.

Еще сложнее для программы работать с типичной журнальной страницей, которая содержит в себе иллюстрации, колонки текста и, возможно, даже таблицы с данными, набранными шрифтами разных размеров и формы. До 1988 года микрокомпьютерные системы ОРС читать такие тексты не могли.

Типографский набор и разделение текста на колонки является обычным делом в полиграфии, так как такие тексты просто удобнее читать человеку. Но они же представляли собой огромную проблему для компьютерного распознавания символов. Лишь совсем недавно появились системы, стоящие дешевле 2000 долларов, которые нормально работают с журнальными текстами.

А всего три года назад ОРС сканеры Kurzweil (Xerox) — самые дешевые тогда программы, способные читать журнальный текст — стоили 10 тысяч американских долларов.

Но с тех пор цены на Kurzweil (Xerox) упали, а на рынке появились изделия фирм Calera и Caere — системы ОРС для IBM и Mackintosh, которые читают типографский текст, обучаются в процессе чтения и даже автоматически анализируют структуру колонок текста на странице. Эти замечательные программы могут сканировать текст, разбитый на 2-3 колонки и автоматически располагать его в файле в нужной последовательности, даже при наличии на странице иллюстраций.

Я лично пользовался программой Caere OmniPage как на Mackintosh, так и на PC с процессором 80386, и нашел, что они обе работают очень хорошо, хотя, как и большинство ОРС-программ, всегда появляются несколько ошибок, которые в зависимости от будущего использования сканированного текста нужно или не нужно будет исправлять. Это очень важный момент, который потенциальные пользователи ОРС часто игнорируют.

Все системы распознавания символов делают ошибки, но эти ошибки не всегда важны настолько, чтобы их исправлять.

Пара примеров для пояснения. Предположим, вы сканируете документы, которые будут входить в состав большой текстовой базы данных.

Если она будет использоваться просто для того, чтобы по ключевым словам найти требуемый документ, который вы потом просто возьмете из нужной папки — в этом случае даже относительно большое число ошибок вполне приемлемо, а использование сканера для создания полного каталога документов гораздо проще найма специального человека для этой цели.

С другой стороны, если после сканирования оригинал будет уничтожен, или если процесс ОРС должен произвести почти совершенную копию исходного документа, то потребуется кто-то, кто прочтет компьютерную версию документа и внесет в нее необходимые исправления для полного соответствия оригиналу.

В последнем случае может оказаться дешевле нанять машинистку, которая просто перепечатает документ вручную. Это особенно верно в случае, когда использование системы ОРС требует постоянного присутствия оператора.

Сканеры

С программами, читающими тексты, может быть использовано несколько типов сканеров — настольный, ручной строчный и ручной страничный.

Настольный сканер имеет плоскую стеклянную поверхность, как у многих ксероксов, а сканирующая головка либо двигается вдоль страницы, либо изображение передается на нее передвигающимся зеркалом.

Эти устройства используются для считывания с газетных страниц или переплетенных книг. Они требуют наличия оператора, который аккуратно переворачивает страницы в процессе сканирования.

Сканеры с перемещением страницы двигают лист бумаги с текстом относительно неподвижной сканирующей головки. В некоторые из них необходимо вставлять страницы по одной вручную, тогда как другие имеют устройство для автоматической подачи страниц из пачки. В последнее время многие настольные сканеры также комплектуются таким механизмом.

Кроме того, существуют два типа ручных сканеров.

Первый — это строчный сканер, который содержит один чувствительный элемент и источник света, передвигаемый пользователем вдоль строки. Этот тип сканеров продавался в очень небольших количествах, но мои испытания показали, что он работает вполне приемлемо.

Строчные сканеры используются для ввода отдельных слов, адресов или чисел непосредственно в обрабатываемый программой текст в месте расположения курсора. Их удобно использовать для копирования графической информации непосредственно в ячейку электронной таблицы или поле базы данных.

К этому типу относятся Soricon DataSweep, а также снятый с производства HandScan фирмы Saba Technologies.

Хотя фирма Saba и прекратила свое существование, я испытал этот сканер и обнаружил, что он дает мало ошибок и удобен, в частности, для бухгалтерских работ.

Другой пример ручного сканера — ScanMan фирмы Logitech. Он перемещается не вдоль строки текста, а сканирует сразу целый столбец шириной 10 сантиметров.

Он гораздо дешевле настольного сканера потому, что в нем нет движущихся частей, а перемещает сканер относительно текста оператор вручную.

Обычно такие сканеры обладают высоким качеством и очень удобны для сканирования колонок текста для программ ОРС или изображений для настольных типографий. Для широких колонок текста у них обычно есть утилиты, позволяющие соединить встык несколько считанных столбцов друг с другом.

Заключение

Собираясь приобретать систему оптического распознавания символов, важно помнить, что даже лучшие из них делают ошибки и что система ОРС даже на быстром микрокомпьютере тратит минимум две минуты на считывание и преобразование одной страницы текста.

Если вы создаете файл в формате определенного текстового процессора, программа распознавания часто будет пытаться преобразовать и такие элементы текста, как подчеркивание, но обычно ни с чем, кроме машинописных текстов, этот режим удовлетворительно не работает.

Прежде чем начать думать о покупке системы ОРС, остановитесь и спросите себя, будет ли она работать быстрее, чем машинистка, особенно в случае, если ваши материалы критичны к ошибкам и требуют многократной проверки путем сличения копии с оригиналом, что практически удваивает время сканирования. Ряд крупных издательств в США предпочли содержание штата машинисток. Это создает дополнительные рабочие места и просто более экономично.

Технология оптического распознавания улучшается каждый день и она, определенно, займет свое место в каждом — даже небольшом — офисе. Но не ожидайте чудес, по крайней мере, сейчас.

Скоро, когда будет достигнут требуемый уровень безошибочного преобразования, ОРС станет обычным рабочим средством. Но уже сейчас в ряде мест, где машинистки дороги или их нет, например, в больших городах, ОРС активно применяется во все большем числе офисов. Многие американские и европейские фирмы перекладывают на ОРС большую часть своей повседневной рутинной работы.

Но поскольку распознавание символов сопряжено все же с определенными проблемами, я бы настоятельно рекомендовал перед покупкой провести обширные тесты для того, чтобы выяснить, будет ли конкретная система удовлетворять ваши нужды.

Ресурсы

У меня нет информации о том, продают ли перечисленные ниже фирмы программы в Советский Союз, но, скорее всего, большинство из них ответит на письменный запрос.

Все цены указаны в американских долларах и не включают в себя цены за пересылку, налоги, и другие сборы, взимаемые даже при продаже внутри Соединенных Штатов. Перечисленные системы распознают только латинский алфавит.

Все данные приводятся чисто в информационных целях. Автор не связан ни с одной из фирм-производителей.

Фирма CompuScan
300 Broadacres Drive
Bloomfield, New Jersey 07003 U.S.A.
201-338-5000

Система PCS 235 — большой страничный сканер с программой ОРС, прекрасно работающий с машинописными текстами, но не читающий типографский набор.

Цена системы 3495 долларов, она должна быть подключена к PC через стандартный последовательный порт RS-232.

Параметры компьютера — скорость, емкость диска — не имеют значения, так как вся обработка проводится в сканере. PCS может обработать до 50 страниц без вмешательства оператора и имеет лучший (из испытанных мной) механизм подачи листов бумаги.

Фирма Intelligent Optics
4 Heritage Park Rd.
P.O. Box 712
Clinton, Connecticut 06413 U.S.A.
203-669-3650

Сканер IOC — это система ОРС, способная обрабатывать стопку листов бумаги. Цены — не ниже 4000 долларов за весь комплект.

Сканер также подсоединяется к PC через последовательный порт, но, в отличие от предыдущей системы, скорость машины начинает играть существенную роль, так как обработка ведется уже силами самого компьютера.

Фирма Xerox Imaging Systems
535 Oakmead Parkway
Sunnyvale, California 94089 U.S.A.

Читающий любой шрифт пакет Acutext для Mackintosh с процессором 68030 продается за 799 долларов.

Более медленный и менее точный, чем OmniPage, тем не менее хорошо работает с изображениями низкого качества. Для их правильной обработки он использует встроенный словарь.

Фирма Calera Recognition Systems
2500 Augustine Drive
Santa Clara, California 95054 U.S.A.
408-986-8006

TopScan для MS-DOS и TrueScan для Mackintosh — полностью автоматические системы ОРС, распознающие типографский набор и размещение текста в несколько колонок. Цена — 2995 долларов.

В виде отдельной системы ОРС продукт стоит 15000 долларов вместе с компьютером и сканером.

Очень мощная программа.

Фирма Kurzweil Computer Products (Xerox)
185 Albany St.
Cambridge, Massachusetts 02139 U.S.A.
617-864-4700

7320 OCR System — сканер ОРС, распознающий типографский набор и колонки текста. Работает под управлением MS-DOS и требует специальной платы-ускорителя. Цена — 5000 долларов без компьютера.

Фирма Soricon
4725 Walnut St.
Boulder, Colorado 80301 U.S.A.
303-440-2800

Строчный сканер DataSweep 2 стоит 2300 долларов. Неизвестно, производится ли он до настоящего времени.

Фирма Caere
100 Cooper Court
Los Gatos, California 95030 U.S.A.
408-395-7000

Пакет OmniPage, автоматически работающий с типографскими текстами и автоматически распознающий набранный в несколько колонок текст, продается за 795 долларов для пользователей Mackintosh. Аналогичный пакет для PC с процессором 80386 или 80486 стоит 895 долларов.

Вместе с дополнительной ускоряющей платой к машинам с процессором 80286 OmniPage стоит 1995 долларов.

Очень хорошая программа. Она не имеет встроенного словаря, а работает, исключительно анализируя форму символов. Для увеличения точности чтения можно использовать ее и со специальным словарем.

Фирма Logitech
6505 Kaiser Drive
Fremont, California 94555 U.S.A.
415-795-8500

Ручной сканер ScanMan с разрешающей способностью от 100 до 400 точек на дюйм для PC продается за 339 долларов. Версия для Mackintosh стоит 399 долларов.

Программа оптического распознавания CatchWord для любого из компьютеров стоит дополнительно 199 долларов.

Фирма New Dest
1015 E. Brokaw Road
San Jose, California 95131 U.S.A.
408-436-200

Сканер PC Scan 2000 для IBM PC или Mackintosh продается за 1395 долларов. Программа распознавания символов Publish Pac — 395 долларов, а плата-ускоритель 795 долларов.

Фирма Olduvai
7520 Red Road, Suite A
South Miami, Florida 33143 U.S.A.

Read-It! — ограниченно обучаемая программа для Macintosh Plus, работающая со многими сканерами и стоящая 495 долларов. Ее улучшенная версия ReadForm, которая может читать и написанные от руки печатные буквы, продается за 695 долларов.

Фирма CTA
747 Third Ave., Third Floor
New York, New York 10017 U.S.A.
212-935-2280

TexPert — обучаемая программа распознавания символов только для латинского алфавита. Работает на Macintosh Plus и более мощных компьютерах этой серии и стоит 995 долларов.

Innovatic
1911 N. Fort Myer Drive, Suite 708
Arlington, Virginia 22209 U.S.A.
703-522-3053

ReadStar II Plus — обучаемая программа ОРС для Mackintosh. Стоит 995 долларов. Очень быстрая.

Фирма Prism Enterprises
14703-E Baltimore Ave., Suite 248
Laurel, Maryland 20707 U.S.A.
301-604-6611

Обучаемая программа TextScan для Mackintosh (395 долларов).

Примечание: несмотря на наличие телефонных номеров звонить по ним НЕ рекомендуется.

Об авторе — Джон Маккормик живет в небольшом угледобывающем городке недалеко от Вашингтона. Он работает с компьютерами с 1963 года и написал за это время более 1500 статей, обзоров, редакционных статей и новостей, относящихся к компьютерной индустрии. Его последняя книга "A Guide to Optical Storage Technology" посвящена оптическим дискам, приводам и другим образцам техники для оптического хранения информации.



Конечно, новое поколение переносных компьютеров, построенных на базе процессора 386SX, нельзя даже сравнивать по возможностям с первыми моделями, однако, как это часто бывает, стремление увеличить мощь и производительность этих машин не могло не сказаться на их размерах и весе, минимизация которых изначально являлась основной целью разработки этого класса устройств. В обзоре мы расскажем о новейших моделях, подчеркнув наряду с несомненными достоинствами появившиеся недостатки, связанные с их "непортативностью".

НОВЫЕ ПЕРЕНОСНЫЕ компьютеры 80386SX

Пользователи портативных компьютеров так же заинтересованы в увеличении производительности компьютеров, как и пользователи обычных персональных компьютеров, но они готовы платить бóльшую цену за возможность "упаковки" производительности настольного компьютера в переносной чехол. Как правило, такие пользователи согласны заплатить больше за портативность даже в ущерб качеству компьютера. Например, хотя тонкие жидкокристаллические или газоплазменные дисплеи могут обеспечить разрешение 640 на 480 точек растра, присущее VGA-мониторам, но вместо сотен цветов VGA-дисплеев они могут обеспечить не более 32 оттенков одного цвета. Жесткие диски портативных компьютеров работают достаточно быстро даже для процессора 80386, но чаще всего их емкость ограничивается 20, в лучшем случае 40 Мбайтами.

Можно, конечно, сказать наоборот: недостатки переносных компьютеров, работающих на базе процессора 80386 при частоте 16 Мгц, полностью компенсируются наличием встроенных портов для подсоединения модемов и принтеров, дисководов для гибких и жесткого дисков и, что более важно, мощью процессора. К тому же, находясь в офисе, вы сможете подключить к этим компьютерам цветной VGA-дисплей и стандартную клавиатуру.

В этом обзоре мы расскажем о четырех переносных компьютерах четырех различных фирм. Все они снабжены процессором 80386SX, все работают на частоте 16 Мгц и, стало быть, обладают примерно одинаковой скоростью. Однако, компьютеры Lap Pro 386SX фирмы Dauphin и System 316LT фирмы Dell снабжены встроенными аккумуляторами, что позволяет использовать их там, куда хватит сил унести, а в компьютерах Lyte-Byte 5400 фирмы Micro Express и 1450SX фирмы GRiD Systems такие аккумуляторы отсутствуют, что по мнению фирм увеличивает их "переносимость", но нести их придется туда, где есть розетки.

Lap Pro 386SX
Фирма Dauphin

Фирма Dauphin Technology была основана в 1987 г. Эта фирма специализируется на поставках систем большим корпорациям и оптовым продавцам, укомплектовывающим продаваемые изделия чем-то своим, конечно, за дополнительную плату, и федеральному правительству, но она также продает переносные компьютеры и конечным пользователям. За 4995 долларов компьютер Lap Pro 386SX поставляется с 2 Мбайтами памяти, 40-Мбайтным винчестером, дисководом для гибких дисков 1,44 Мбайт, дисплеем стандарта EGA на жидких кристаллах с подсветкой и существенным набором программного обеспечения, включающим операционную систему DR DOS 3.41 фирмы Digital Research.

Несколько слов об особенностях этого компьютера. Ряд проблем связан с нестандартной 75-клавишной клавиатурой. Во-первых, клавиши управления курсором расположены над основной областью алфавитно-цифровых клавиш, что весьма неудобно, поскольку приходится далеко тянуться. Во-вторых, почему-то отсутствуют функциональные клавиши, и, ежели таковые требуются для вашего программного обеспечения, то приходится одной рукой нажимать клавишу FUNC, а другой — цифровую клавишу с соответствующим номером. В-третьих, порой при быстром вводе информации на экране вдруг появляются дополнительные символы, эффект напоминает одновременное нажатие трех разных клавиш. И наконец, в-четвертых, клавиши ESC и / находятся в крайне неожиданных местах, к чему непросто привыкнуть человеку, работавшему прежде на нормальной клавиатуре.

На 10-дюймовом экране компьютера синие символы высвечиваются на светло-зеленом фоне. В общем, конечно, разобрать, что выдано на экран, можно, но на многих других жидкокристаллических дисплеях с подсветкой картинка выглядит гораздо четче. Хотя дисплей обеспечивает разрешение только 640 на 400 точек растра, его видеосхемы поддерживают при подключении внешнего монитора разрешение 640 на 480 — то есть, стандарт VGA.

Корпус компьютера неплохо смотрится внешне, но, как и клавиатура, обладает рядом недостатков. Так, вмонтированная ручка недостаточно удобна, чтобы распределить равномерно вес компьютера (7,5 кг), что делает весьма неудобной его переноску на большие расстояния. Разъемы, находящиеся на задней стороне, не имеют никакой защиты и, при случайном падении компьютера, скорее всего будут сломаны. Да и вообще, довольно сложно открыть этот компьютер, не сломав ногтей.

Среди несомненных достоинств компьютера следует отметить высоконадежный 40-ваттный источник питания, который позволяет использовать для подключения компьютера как стандартные розетки 100 или 220 В, так и его собственные аккумуляторы или 12-вольтовый внешний источник питания, например, автомобильный адаптер для прикуривателя.

Хотя вскрытие корпуса компьютера автоматически приводит к прерыванию его гарантии, мы сделали это, чтобы сообщить читателям, что интересного у него внутри. Так вот, на системной плате имеется место для увеличения памяти до 4 Мбайт, кроме того, можно установить сопроцессор, например, 80387SX, а также подключить внешнюю клавиатуру, внешний дисковод для гибких дисков 360 Кбайт или 1,2 Мбайт и установить внутри модем, работающий со скоростью 2400 бод.

Этот компьютер поставляется с таким набором программного обеспечения, что его практически сразу можно использовать для любых задач. DR DOS — это операционная система, совместимая с MS-DOS, включающая тот же набор несколько модифицированных команд. Интегрированный пакет Alphaworks фирмы Alpha Software включает текстовый процессор, СУБД, электронную таблицу, программу проверки правописания, словарь, деловую графику и программу связи. Кроме того, в этот набор включена утилита передачи файлов Laplink фирмы Traveling Software.

Фирма Dauphin Technology обеспечивает годовую гарантию на всю систему. В случае выхода из строя системы в течение гарантийного периода фирма бесплатно установит у вас аналогичную систему на время ремонта старой, если этот ремонт не может быть завершен в течение 48 часов.

System 316LT
Фирма Dell Computer

Наконец появился первый переносной компьютер фирмы Dell. Его ждали долго и, судя по всему, не зря. Базовая конфигурация поставляется с 1 Мбайтом памяти, 20-Мбайтным винчестером фирмы Conner Peripherals, дисководом для 3,5-дюймовых гибких дисков и дисплеем стандарта VGA. В комплект, стоящий 3499 долларов, входит также аккумулятор, позволяющий использовать компьютер без внешнего питания. Если добавить программное обеспечение MS-DOS 3.3 и нарастить память до 2 Мбайт, то цена увеличится до 3798 долларов, при замене винчестера на 40-Мбайтный придется заплатить 4000 долларов. В этом компьютере хорошо продумана защита от внешних воздействий: гнезда видеовыхода и клавиатуры, последовательный и параллельный порты и два гнезда для возможного подключения внешних устройств закрыты пластиной, которая, в случае необходимости, скользя, убирается под корпус компьютера. Этот компьютер весьма компактен, что можно сказать и о его 83-клавишной клавиатуре, хотя клавиши имеют обычный размер. Клавиши управления курсором расположены вдоль правого края клавиатуры, цифровая группа включена в общую алфавитную часть клавиатуры и для ее использования следует нажать клавишу Shift. Никаких проблем с вводом информации обнаружено не было.

Можно сказать несколько добрых слов о дисплее этого компьютера: он достаточно ярок, четок и хорошо интерпретирует цвет оттенками серого. В комплект программного обеспечения при поставке включены резидентная программа для улучшения качества графических изображений и утилита установки видеорежимов (видеоконтроллер поддерживает стандарты EGA, CGA, VGA, а также монохромный режим). Если снять жидкокристаллический дисплей, то можно подключить внешний монитор VGA.

Многие изготовители переносных компьютеров негативно относятся к вскрытию компютера, но не фирма Dell! В руководстве подробно рассказано, как снять корпус — для этого надо лишь отвинтить девять винтов — и как нарастить память до 8 Мбайт, подключить дисководы, сетевую плату, плату факсимильного устройства, модем и выполнить прочие модификации.

Еще одним плюсом этого компьютера являются съемные аккумуляторы. Эти аккумуляторы размером с сигаретную пачку легко вставляются и снимаются. Компьютер поставляется с двумя аккумуляторами, один из которых является резервным. Каждого акуумулятора хватает на два часа работы, за 10 минут до истечения срока службы аккумулятора компьютер издает предупреждающий сигнал и уменьшает скорость до 8 Мгц. Небольшая резервная батарея поддерживает работу компьютера в течение двух минут, давая возможность произвести замену аккумулятора без выключения компьютера.

Единственным недостатком этого компьютера (впрочем, если это можно назвать недостатком) являются размеры источника питания переменного тока, одновременно являющегося устройством зарядки аккумуляторов. Если Вы уезжаете надолго и собираетесь использовать свой 316LT более 4 часов, то Вам придется взять с собой на пару обуви или на буханку хлеба меньше: именно столько места занимает зарядное устройство. Но, проигрывая в размерах, выигрываем в силе: это устройство полностью заряжает аккумулятор всего за 3 часа, тогда как другим аналогичным устройствам для этого требуется 8 часов.

Фирма дает на этот компьютер годовую гарантию, за дополнительную плату вы можете увеличить срок обслуживания компьютера до 4 лет.

Подводя итог, можно сказать, что характеристики и вес (6,8 кг) этого компьютера позволяют поставить его на одну полку с последними компьютерами фирм Toshiba и Zenith — признанными лидерами в этой области — хотя срок службы аккумуляторов последних составляет более 3 часов. Зато 316LT стоит почти на 2 тысячи долларов дешевле.

Lyte-Byte 5400
Фирма Micro Express

Фирма Micro Express со дня основания, то есть с 1986 г., продает широкий спектр переносных, портативных и настольных компьютеров, в настоящее время в этот круг входят компьютеры, начиная с класса 286 и кончая 486. Поскольку фирма значительную часть своей продукции поставляет оптовым продавцам, вносящим в нее свои модификации, впрочем, весьма незначительные, компьютер Lyte-Byte 5400 может встретиться под маркой LT5400 фирмы Chicony Electronics. Этот компьютер не имеет автономного питания от аккумуляторов, во всем остальном он весьма неплох, к тому же за невысокую цену. Базовая конфигурация с 1 Мбайтом памяти и 40-Мбайтным диском стоит 2999 долларов. В эту цену входит переносная сумка с плечевым ремнем. При добавлении MS-DOS 3.3 цена увеличивается до 3074 долларов.

Внешне довольно неуклюжий корпус обеспечивает достаточную защиту параллельного и последовательного портов и разъемов, к которым могут быть подсоединены внешняя клавиатура, аналоговый или мультичастотный монитор и внешний дисковод для гибких дисков 5,25 дюймов.

85-клавишная клавиатура представляет собой удачную версию стандартной клавиатуры AT с группой цифровых клавиш, расположенных справа от алфавитной области, и десятью функциональными клавишами, находящимися в верхней части клавиатуры. Клавиша Fn используется в сочетании с другими клавишами для изменения скорости компьютера, переключения между существующим газоплазменным и внешним дисплеями и переключения дисплея в режим супер-VGA.

Графика и текст выводятся на экран оранжевым цветом на черном фоне. Цвета передаются 16 оттенками серого. Дисплей имеет собственный выключатель питания, поэтому его можно отключить при работе с внешним монитором. 8-разрядная плата VGA вместе с BIOS фирмы Trident Microsystems поддерживают VGA и все предыдущие видеостандарты, включая еще и Геркулес и супер-VGA с разрешением 800 на 600 точек растра.

В отличие от большинства переносных компьютеров, конструкция Lyte-Byte позволяет легко отсоединить клавиатуру, чтобы получить доступ к системной плате для установки сопроцессора 80387SX, дополнительной памяти (до 4 Мбайт) и изменения в случае необходимости положения микропереключателей, определяющих характеристики компьютера.

40-Мбайтный жесткий диск обладает автопарковкой головок и обеспечивает высокую скорость передачи данных, совпадающую с возможностями процессора. Источник питания является самонастраивающимся на напряжение от 95 до 270 В, что очень удобно при путешествиях в разные страны.

Фирма Micro Express обеспечивает на этот компьютер 15-месячную гарантию и ремонт в течение 48 часов с оплатой фирмой транспортировки компьютера. Если пользователь может обойтись без автономного питания, то этот компьютер при его характеристиках и цене является одним из наилучших в своем классе.

1450SX
Фирма GRiD Systems

GRiD Systems, дочерняя фирма компании Tandy, тоже считает, что для компьютеров этого класса портативность важнее, чем автономное питание. Ее модель 1450SX является, вероятно, самой компактной и легкой из построенных на процессоре 386SX — благодаря отсутствию встроенных аккумуляторов она весит всего 3,6 кг. Базовая конфигурация 1450SX работает только от сети, но возможности машины могут быть расширить: за 25 долларов можно купить адаптер для подключения компьютера к автомобильному прикуривателю, за 299 долларов — аккумуляторы, которых при весе 1,36 кг хватает на 2 часа работы, или за 399 долларов — аккумуляторы весом 1,8 кг, которых хватит на 4 часа. Аккумуляторы легко подключаются к гнезду, находящемуся на нижней части компьютера.

В базовую конфигурацию компьютера входят 1 Мбайт оперативной памяти, расширяемой до 5 Мбайт, дисковод для гибких дисков 1,44 Мбайт, 10-дюймовый дисплей стандарта VGA (черное изображение на белом фоне) с разрешением 640 на 480 точек и 32 оттенками серого, 20-Мбайтный винчестер фирмы PrairieTek (фирма GRiD планирует перейти вскоре на 40-Мбайтные диски после завершения фирмой PrairieTek их разработки) и полноразмерную 82-клавишную клавиатуру. 1450SX имеет порт для подключения внешнего VGA-монитора, но не может работать одновременно со своим монитором и внешним дисплеем.

Цена базовой конфигурации — 4995 долларов — существенно ниже, чем у аналогов конкурирующих фирм Toshiba и Zenith, а при оптовых закупках может быть снижена до 3500 долларов, что позволяет поставить эту модель на одну полку с новым компьютером фирмы Dell.

Ниже мы приводим краткие характеристики еще нескольких переносных компьютеров этого класса. Общей особенностью этих компьютеров является то, что их конфигурация может быть существенно расширена или сужена, исходя из потребностей покупателя, поэтому мы не указываем их стоимость, тем более, что читатель уже понял порядок цен на предыдущих примерах.

1100LX
Фирма Acer Technologies

Вес 7,25 кг (с аккумуляторами). Время непрерывной работы на батареях: 3 часа. Дисплей: монохромный жидкокристаллический с подсветкой. Винчестер: 40 Мбайт. Оперативная память: до 5 Мбайт. Клавиатура: полноразмерная, отсоединяемая. Интерфейсы: внутреннее гнездо для модема, шесть внешних портов для подключения дисковода для гибких дисков 5,25 дюймов, цифровой клавиатуры, принтера, мыши, цветного VGA-монитора или блока расширения с тремя AT-совместимыми гнездами и свободным местом для жесткого диска и стримера. Встроенный модем (2400 бод), сопроцессор, цифровая клавиатура и внешний флоппи-дисковод могут быть поставлены дополнительно.

M316
Фирма Olivetti

Вес 6,7 кг. Размеры: 9,9x33x36 см. Время непрерывной работы на батареях: от 2 до 2,5 часа. Встроенное зарядное устройство заряжает батареи за 5 часов. Дисплей VGA монохромный жидкокристаллический. Размеры экрана 21,25x16,25 см. Винчестер 20 или 40 Мбайт. 3,5-дюймовый накопитель на гибком диске 1,44 Мбайта. Оперативная память от 1 до 4 Мбайт на системной плате. Клавиатура эмулирует функции 101/102 клавиш. Сопроцессор и выносная цифровая клавиатура поставляются дополнительно. Компьютер оснащен параллельным и последовательным портами, портом для мыши (PS/2), 16-разрядным AT-совместимым гнездом расширения и разъемами для подключения внешнего накопителя и модема.

Prospeed 386SX laptop
Фирма NEC Canada

Вес 6,8 кг. Размеры: 32,25x8,65x37,25 см. Время непрерывной работы на батареях: 3 часа. Дисплей VGA монохромный жидкокристаллический с подсветкой. Винчестер 40 или 100 Мбайт. 3,5-дюймовый накопитель на гибком диске емкостью 1,44 Мбайта. Оперативная память от 1 до 4 Мбайт на системной плате (возможно расширение до 16 Мбайт). Клавиатура содержит 82 клавиши, включая 12 функциональных, и клавиши управления курсором. Компьютер оснащен последовательным и параллельным портами, разъемами для подключения модема и внешнего дискового накопителя.

LT/386
Фирма Tandon

Вес 6,6 кг. Размеры 31,75x34x8,5 см. Время непрерывной работы на никелево-кадмиевых батареях: от 2 до 3 часов. Встроенный сетевой адаптер выполняет также роль зарядного устройства. Дисплей EGA или VGA монохромный жидкокристаллический. Винчестер 40 Мбайт. 3,5-дюймовый накопитель на гибком диске 1,44 Мбайта. Оперативная память от 1 до 5 Мбайт на системной плате. 82-клавишная клавиатура эмулирует функции 101/102 клавиш. Компьютер оснащен параллельным и последовательным портами, а также 16-разрядным AT-совместимым гнездом расширения.

LP/SX
Фирма DATA

Вес 6,7 кг. Газоплазменный дисплей VGA с разрешением 640x480 точек (16 полутонов). Разъем для подключения цветного монитора VGA. Оперативная память 1 Мбайт (до 4 Мбайт на дополнительной плате). Винчестер 40 Мбайт с временем доступа 25 мс. 3,5-дюймовый накопитель на гибком диске 1,44 Мбайта. Клавиатура содержит 87 клавиш. Компьютер оснащен одним параллельным и двумя последовательными портами, разъемами для подключения дополнительной клавиатуры и 5,25-дюймового накопителя на гибких дисках.

Б.Молчанов

По материалам:

J.Helliwell. “The Dell 386SX laptop is one heavyweight worth waiting for“. PC/Computing, April, 1990.
F.Paul. “GRiD's light approach to SX“. PC/Computing, April, 1990.
V.Wood. “Guide to portable PCs“. Office Equipment & Methods, May, 1990.
B.Machrone. “Laptop confusion everywhere“. PC Magazine, May 15, 1990.
M.Jarvela, D.Rowell, J.Wolfskill. “Power plus portability: three travelling 386SXs“. PCResource, June, 1990.



Наш журнал продолжает публикации о приемах грамотной работы на персональном компьютере. В этом выпуске КомпьютерПресс мы расскажем об использовании утилиты операционной системы RECOVER, о том, как правильно пользоваться утилитой CHKDSK, об аттрибутах файлов и методах работы с ними.

Работаем грамотно

ПРАКТИЧЕСКИЕ СОВЕТЫ ПО МИНИМИЗАЦИИ УЩЕРБА В СЛУЧАЕ ПОЯВЛЕНИЯ НА ДИСКЕ ПЛОХИХ СЕКТОРОВ (ПРИМЕНЕНИЕ ПРОГРАММЫ RECOVER)

Каждый пользователь персонального компьютера живет в страхе получить на экране такие сообщения, как "Data error" — "Ошибка данных" или "Sector not found" — "Сектор не найден". В лучшем случае это означает, что какой-то один файл испорчен в результате появления на диске плохого сектора. Это же может означать, что пострадали несколько файлов.

В такой ситуации помочь спасти файл может программа RECOVER операционной системы DOS. Но если вы неправильно примените эту программу, то рискуете доставить себе дополнительные хлопоты на несколько часов, а то и дней.

Общий термин "плохой сектор" применим ко всем тем дисковым секторам, которые DOS не может надежно читать или записывать. Причины возникновения плохих секторов разнообразны: от неправильного обращения с дискетами до неисправности оборудования. Поскольку программа FORMAT улавливает большинство дефектов при подготовке диска к работе, последующие сообщения о плохих секторах указывают на появление таких секторов уже после форматирования.

Ниже приводится ряд советов по минимизации ущерба от появления плохого сектора.

1. Пытайтесь! Пытайтесь! Пытайтесь! При появлении обычного сообщения "Abort, Retry, Fail, or Ignore" нажмите "R" как минимум трижды, пытаясь повторить обращение заново. Если результат не меняется, то мы вас поздравляем — проблема налицо. Если, отчаявшись спасти все, вы желаете спасти хоть какую-нибудь оставшуюся информацию, нажмите "I": DOS проигнорирует ошибку и запишет на диск все те части файла, какие он сможет записать. Это спасет текстовый файл, но погубит программный.

2. Попытайтесь сохранить файл заново. Если ошибка появилась в момент сохранения электронной таблицы или документа, попытайтесь спасти файл заново с другим именем или на другом дисководе. Таким образом, DOS разместит "новый" файл в другом месте. При работе с такими программами, как WordStar, которые не позволяют менять дисковод или имя файла, воспользуйтесь командой записи блока, которая позволит поместить весь документ в новый файл. К сожалению, вы не можете делать этого при работе с программами бухгалтерских расчетов или базами данных, которые записывают лишь измененную часть информации. Плохой сектор в этом случае остается в файле.

3. Выключите и повторно включите компьютер. Есть шансы, что перезагрузка компьютера "вылечит" сектор, временно оказавшийся плохим. После перезагрузки попробуйте снова поработать с файлом. Если DOS продолжает сообщать о наличии ошибки, с фай-

лом действительно случилось что-то серьезное, и настало время перейти к следующему шагу.

4. Запустите для этого файла программу RECOVER. Программа RECOVER работает как с гибкими, так и с жесткими дисками, перенося все, что в ее силах, в новую копию файла. Ее следует применять с осторожностью (см. пункт 8). Те части информации, которые невозможно извлечь из плохого сектора, заменяются пробелами. Программа RECOVER также отмечает плохие сектора в таблице размещения файлов (FAT) диска, чтобы они в дальнейшем не использовались для новой информации.

5. Отредактируйте восстановленные текстовые файлы; будьте готовы распрощаться с текущими версиями файлов других типов. Большинство документов и все файлы данных, содержащие только текст в формате ASCII, можно путем редактирования привести к исходному виду. Разумеется, в том случае, когда вы в состоянии вспомнить потерянную информацию. Следует удалить лишние пробелы и заново ввести в файл его недостающие части.

Любой восстановленный программный файл нужно заменить его оригиналом или резервной копией. Запускать восстановленные версии программных файлов — безумие.

Другие файлы (такие как электронные таблицы и базы данных) имеют шансы быть спасенными. Попробуйте вызвать их своими программами. Если вам повезет, вы сможете ввести потерянную информацию заново. Если ваша программа на что-то жалуется или ведет себя странным образом, воспользуйтесь резервными копиями, но сохраните под рукой восстановленную версию файла. При помощи команд TYPE или DEBUG вы сможете просмотреть ту текущую информацию, которая отсутствует в резервной копии.

6. Чаще делайте полные резервные копии диска. То, что послужило причиной первого появления плохого сектора, может заявить о себе снова. Если за пару недель плохих секторов больше не появится, вы можете ослабить график резервного копирования.

7. Повторение ситуации — показатель серьезной неисправности. Вторичное появление плохих секторов может быть следствием неисправности дисковода. Если плохие сектора появились на двух дискетах или в двух разных местах одного жесткого диска, следует выполнить диагностику дисковода.

8. Избегайте использования программы RECOVER для всего диска. Запомните: нельзя использовать программу RECOVER для всего диска, за исключением случая, когда плохой сектор появился в корневом каталоге. При запуске программы RECOVER для всего диска обрабатываются даже неповрежденные файлы и подкаталоги. При этом вы потеряете исходные имена, даты и размеры файлов. Программа RECOVER размещает каждый файл в корневом каталоге. При этом файлу присваивается имя FILEnnnn.REC (где nnnn — четырехзначное число). Этот процесс продолжается до тех пор, пока в корневом каталоге остается свободное пространство.

Синтаксис этой команды следующий:
RECOVER [d:][путь][имя файла] , где
d: — имя дисковода;
путь — имя каталога;
имя файла — имя того файла, который следует восстановить (использовать символы "*" и "?" не разрешается).

В некоторых изданиях документации к системе DOS 4.0 содержится ошибка: вы не можете запускать программу RECOVER для одного подкаталога. Возможны лишь два варианта: либо восстановление одного файла, либо восстановление всего диска. Программа RECOVER надежно работает при восстановлении отдельного файла. Результат запуска программы для всего диска может оказаться печальным; в большинстве случаев лучше предпочесть этому резервное копирование с последующим переформатированием диска.

ХОРОШИЙ СПОСОБ ПОДДЕРЖАНИЯ ЛОГИЧЕСКОГО ЗДОРОВЬЯ ВАШЕГО ЖЕСТКОГО ДИСКА (ПРИМЕНЕНИЕ ПРОГРАММЫ CHKDSK)

Большинство пользователей время от времени запускает системную программу проверки диска CHKDSK, чтобы посмотреть, сколько осталось свободного дискового пространства или сколько доступно оперативной памяти. Однако программа CHKDSK может обеспечить вас и другой, гораздо более полезной информацией, необходимой для поддержания здоровья компьютера и, кроме того, выполнить необходимые действия.

В какие моменты особенно важно запускать команду CHKDSK? Сразу после внезапного сбоя программы или зависания системы. Но прежде чем браться за дело, удостоверьтесь, что вы правильно понимаете отличие команды CHKDSK с ключом /F от команды CHKDSK без ключа. Ключ /F предназначен для исправления логических ошибок в организации данных на диске. Никогда не запускайте программу CHKDSK с ключом /F, не запустив ее предварительно без ключа. Вызов программы CHKDSK сразу после сбоя облегчает решение проблем, однако вызов этой программы с ключом /F в ряде случаев может лишь ухудшить положение.

Программа CHKDSK пытается обнаружить рассогласование между таблицей размещения файлов (FAT) на диске, записями каталога и фактически хранящимися на диске данными. Обычно причинами логических разрушений являются:

1) сбой питания,

2) перезагрузка компьютера в момент записи на диск,

3) неадекватное поведение программы,

4) ввод пользователем ответа "Abort" или "Fail" после сообщения системы о важной ошибке,

5) нажатие комбинации клавиш Ctrl-Break в момент записи на диск.

Наиболее распространенная логическая проблема состоит в появлении на диске потерянных кластеров — частей файла, которые система записала на диск, но не объявила их принадлежащими этому файлу. Об обнаружении подобной ситуации программа CHKDSK сообщает, например, так: "10 lost clusters found in 3 chains. Convert lost chains to files (Y/N)?" ("В трех цепочках обнаружены десять потерянных кластеров. Преобразовать потерянные цепочки в файлы (ДА/НЕТ)?"). Числа в таком сообщении могут варьироваться, смысл же его в том, что часть информации на диске оказалась преданной забвению.

Существование потерянных кластеров вызывает две проблемы: наличие неполных файлов и необоснованная трата дискового пространства. Поэтому эту ошибку в логической организации следует устранить. Программа CHKDSK будет выявлять потерянные кластеры независимо от того, задали вы ключ /F или нет. Она всегда будет спрашивать, нужно ли преобразовывать потерянные цепочки в файлы. Если вы не указали ключ /F и ответили на вопрос положительно, программа CHKDSK сообщит, какие действия она могла бы совершить, однако фактического исправления диска не произведет. При отрицательном ответе на вопрос программа CHKDSK просто прервет свою работу.

Зачем нужно запускать CHKDSK дважды

Помимо потерянных кластеров программа CHKDSK исправляет ряд других логических проблем. Именно поэтому нужно проявлять осторожность при запуске программы с ключом /F. При задании ключа /F программа CHKDSK может создать новые файлы, затерев ими информацию, которую можно было бы восстановить другим способом.

Следующая логическая проблема, которую способна решить программа CHKDSK, состоит в том, что возможна ситуация, когда нескольким файлам принадлежит одна и та же область на диске. Программа CHKDSK сообщит имена пересекающихся файлов. Вам необходимо скопировать эти файлы на другой диск и удалить оригиналы. Затем изучить копии файлов при помощи текстового редактора или программы, которая их создавала, пытаясь определить, какой из файлов содержит ошибочную информацию. Задача усложняется, если файлы не текстовые.

Иногда сбой программы или системы может временно нарушить нормальное функционирование системы. В этом случае запуск программы CHKDSK с ключом /F может привести к тому, что CHKDSK попытается исправить диск, с которым на самом деле все в порядке, и этим породит реальную проблему. Если программа CHKDSK сообщает о множестве ошибок, перезагрузите систему и запустите CHKDSK заново. Если появились те же сообщения, значит проблемы реально существуют. Если большинство или все сообщения пропали, значит в системе были временные повреждения, которые исчезли при перезагрузке.

После того, как все исправления, которые можно было сделать вручную, выполнены, запустите программу CHKDSK с ключом /F для обнаружения потерянных кластеров. После получения от вас положительного ответа на ее запрос, программа преобразует все группы потерянных кластеров — цепочки — в файлы в корневом каталоге с именами FILEnnnn.CHK (где nnnn — число от 0000 до 9999).

При отрицательном ответе потерянные кластеры объявляются свободными. Это увеличивает свободное пространство на диске, но делает почти невозможным восстановление какой-либо полезной информации.

При положительном ответе вам еще придется провести некоторую исследовательскую работу. Перейдите в корневой каталог и при помощи команды TYPE или текстового редактора просмотрите файлы с именами FILEnnnn.CHK. Изучите также файлы данных, которые вы использовали в момент сбоя программы или системы. Эта работа поможет определить, к каким файлам относится потерянная информация.

Если файлы с именами FILEnnnn.CHK содержат текст в формате ASCII, вы сможете воспользоваться текстовым редактором, чтобы вернуть исходным файлам потерянную информацию. Если же данные представлены в двоичном формате (на экране их изучать бессмысленно), придется воспользоваться специальной программой, например Norton Utilities или Mace Gold.

В некоторых случаях бывает проще удалить файлы, созданные программой CHKDSK, и ввести данные заново. Иногда потерянные кластеры содержат информацию из временных файлов, созданных прерванной программой. Определив, что эта информация вам не нужна, вы можете удалить соответствующие файлы.

ПРИЕМЫ ЗАЩИТЫ ФАЙЛОВ (ПРИМЕНЕНИЕ КОМАНДЫ ATTRIB)

Некоторые ваши файлы более других подвержены опасности случайного стирания, однако последние версии операционной системы DOS обеспечивают удобный способ их защиты.

В группу риска попадают самые важные файлы системы: COMMAND.COM, CONFIG.SYS и AUTOEXEC.BAT. Многие программы установки переписывают два последних файла либо делают в них нежелательные изменения.

Еще одну опасность создает ситуация, когда программные файлы размещаются в одном каталоге с файлами данных. Нежелательно, например, хранить

файлы документов в одном каталоге с текстовым процессором. В один прекрасный день вы, желая стереть некоторые документы, ошибетесь с именами и сотрете сам текстовый процессор.

Начиная с версии DOS 3.0, вам предоставляется возможность защищать любые файлы от стирания и изменений при помощи команды ATTRIB. Эта команда позволяет работать с двумя из шести информационных битов, которые DOS отводит для атрибутов каждого файла. Эти шесть битов определяют, можете ли вы использовать или даже видеть файл.

Два атрибута предназначены для специальных файлов системы DOS: подкаталогов и меток (электронное имя диска). Еще два атрибута управляют системными и скрытыми файлами. Они могут запретить отображение файла при нормальном поиске в каталоге (например, командой DIR). В общем случае вы не можете прямо использовать, копировать или стирать скрытые файлы. Команда ATTRIB имеет дело с последними двумя атрибутами — архивирования и защиты от записи.

Атрибут архивирования включается системой, когда вы создаете или изменяете файл. Он говорит вам либо программе резервного копирования о том, что файл новый или был изменен с момента создания резервной копии и, следовательно, нужно внести его в новую резервную копию.

С точки зрения защиты файлов нас будет интересовать только атрибут защиты от записи. Когда этот атрибут установлен, файл нельзя ни изменить, ни стереть. В остальном файл выглядит как обычно: вы видите его имя в каталоге, можете его копировать, изучать его содержимое. Однако при попытке что-нибудь стереть или записать в нем DOS отреагирует сообщением: "Access denied" — "В доступе отказано".

Применение команды ATTRIB

Синтаксис команды ATTRIB отличается от синтаксиса большинства команд операционной системы DOS. Здесь для изменения атрибутов вам не нужно пользоваться символом "/". Вместо этого используется знак "+" для включения атрибута и знак "-" для выключения. Сразу за знаком следует буква, идентифицирующая атрибут ("R" — для атрибута защиты от записи, "A" — для атрибута архивирования). В отличие от большинства команд системы DOS установка атрибута следует перед именем файла.

Команда ATTRIB может также отображать состояние атрибутов файла. При запуске команды без установки атрибутов (+R, -R, +A или -A) она отобразит имена файлов, перед которыми будет стоять буква "R", если установлен атрибут защиты от записи, и буква "A", если установлен атрибут архивирования.

Чтобы защитить три файла, которые система всегда ищет при загрузке, введите следующие три команды:

```
ATTRIB +R COMMAND.COM
ATTRIB +R CONFIG.SYS
ATTRIB +R AUTOEXEC.BAT
```

Кроме того, установите атрибут защиты от записи для всех программных файлов, например, для текстового процессора и его главного словаря проверки правописания. Это особенно важно сделать, если программные файлы находятся в одном каталоге с файлами данных.

Однако будьте осторожны с файлами, содержащими программные установки. Например, редактор Microsoft Word хранит установки в файле MW.INI. Если вы защитите этот файл от записи, вы не сможете менять установки редактора. То же относится к дополнительным словарям. Защитив эти файлы от записи, вы не сможете добавлять новые слова.

Нужно сказать о ключе /S, который задает обработку файлов в подкаталогах (для версий DOS 3.3 и выше). Это позволит вам за один раз изменить атрибуты многих файлов, даже всего диска.

Ниже приводится пара пакетных файлов, использовать которые проще, чем запоминать ключи команды ATTRIB. Первый файл READONLY.BAT предназначен для включения атрибута защиты от записи для файла или для нескольких файлов (если использовать в именах символы "*" и "?"):

```
ATTRIB +R %1 %2
ATTRIB %1
```

Вторая программа READWRIT.BAT выключает атрибут защиты от записи:

```
ATTRIB -R %1 %2
ATTRIB %1
```

Обе программы меняют установку атрибута, а затем отображают имена файлов с указанием значения атрибута.

Несколько примеров использования ATTRIB.

Чтобы установить атрибут защиты от записи для файла MYFILE.TXT, следует ввести следующую команду:

```
ATTRIB +R MYFILE.TXT
```

Чтобы выключить атрибуты защиты от записи и архивирования для всех файлов в каталоге C:\WORD и его подкаталогах (для версий DOS 3.3 и выше), следует ввести следующую команду:

```
ATTRIB -R -A C:\WORD\*.* /S
```

О.Липкина, О.Моторная

По материалам журнала PC/Computing.

Черный конверт легко скользнул в щель на передней панели, щелкнул флажок и ваша машина готова к диалогу. Как же работает дисковод для гибких дисков — устройство, без которого трудно представить себе современный персональный компьютер?

Дисководы для гибких дисков

Сначала немного истории. Лет 50-60 назад в магнитной записи применялись не компакт-кассеты, а металлические катушки, на которые была намотана стальная проволока. Потом появилась магнитная лента — пластиковая пленка, покрытая окисью железа. В те времена магнитофоны использовали только для записи звука и, вплоть до выпуска первых цифровых вычислительных машин, никому не приходило в голову записывать на магнитную ленту что-нибудь, кроме музыки или речи. Но вот магнитофон подключили к компьютеру и по кабелю понесся поток цифр. Так появились первые огромные лентопротяжки-шкафы с катушками, вмещавшими несколько километров ленты.

Время шло, компьютеры становились все быстрее, а ожидание загрузки данных с ленты все утомительнее. Но тут вспомнился принцип грампластинки — спиральная канавка на пластиковом диске. А почему бы на поверхность диска не нанести магнитный порошок? Тогда для того, чтобы добраться до определенной записи, достаточно установить головку в нужное место и начинать считывать. Только одна спиральная дорожка трансформировалась в сотню кольцевых. Устройство получило название "магнитный диск". Долгое время он оставался сложным, громоздким и дорогим устройством, правда сравнительно быстрым и достаточно емким.

С появлением мини-компьютеров возникла потребность в удобных компактных накопителях. И вот сделан следующий шаг: магнитный слой нанесли на тонкую гибкую основу, подобную той, что используется в магнитной ленте. Чтобы не помять диск, не поцарапать и не испачкать его нежной поверхности, его поместили в достаточно жесткий пластиковый конверт, внутри которого он мог свободно вращаться. Диск стал настолько плоским и легким, что его можно было потерять между журнальных страниц. Соответсвенно уменьшились и дисководы.

Первые гибкие диски (флоппи-диски) имели диаметр 8 дюймов (203мм) и емкость до Кбайт. По мере развития технологии изготовления магнитных слоев и совершенствования самих накопителей, размер диска уменьшался, при этом емкость его увеличивалась. Следом за 8-дюймовыми появились диски диаметром 5,25 дюйма (133мм). Первоначально на них можно было записать всего 160 Кбайт, затем магнитный слой стали наносить на пластиковую основу с обеих сторон и емкость удвоилась, в соответствии с этим удвоилось и количество головок у дисковода. Через некоторое время головки претерпели изменения, позволившиее на одной стороне уместить 80 дорожек при повышенной плотности записи, вместо 40 — при обычной. Емкость диска достигла 1200 Кбайт.

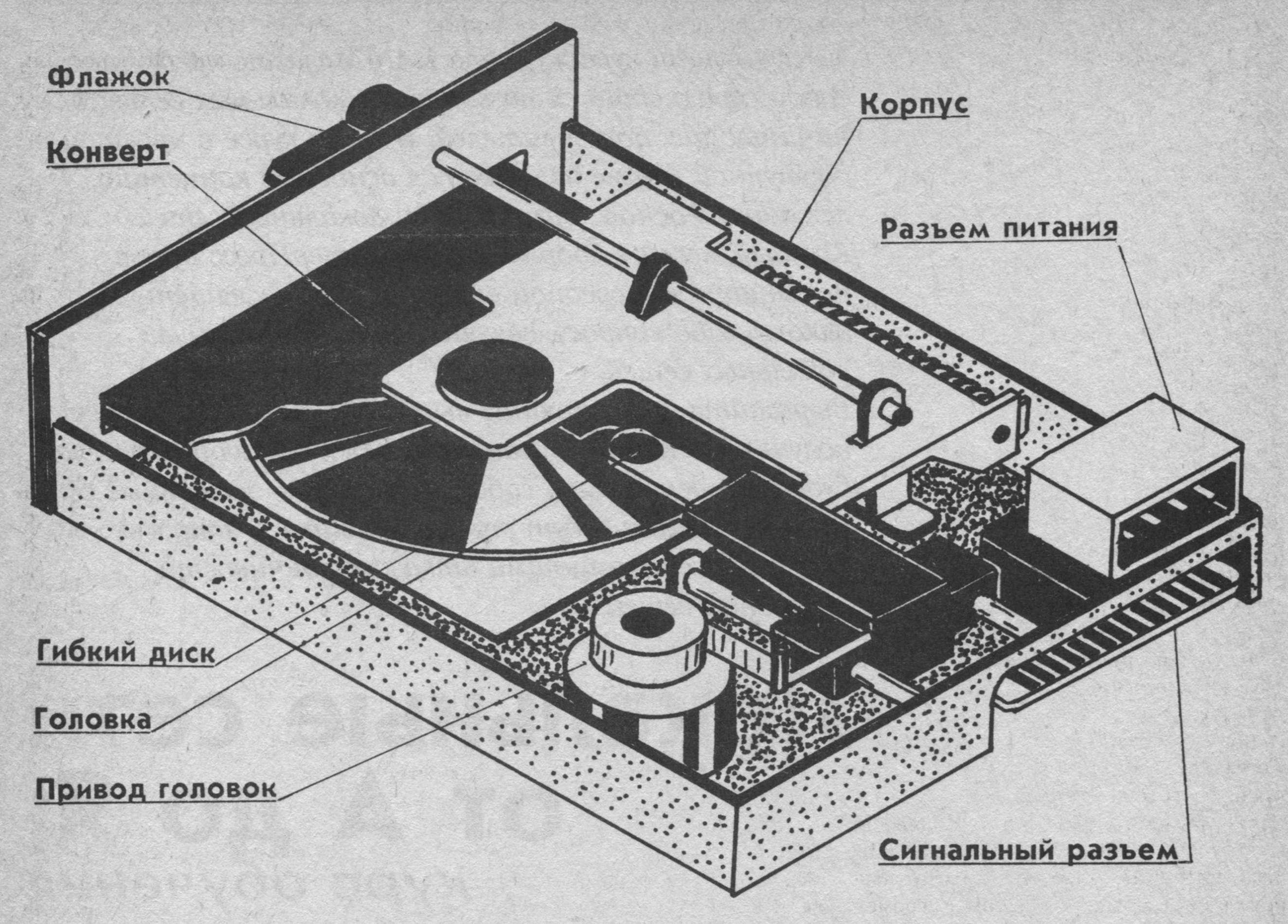
Следующим этапом стали дискеты диаметром 3,5 дюйма (89мм). Емкость этих малюток составляет уже 1440 Кбайт. При такой плотности записи защита магнитного слоя становится особенно актуальной, поэтому диск спрятан в прочный пластмассовый корпус, а зона контакта головок с его поверхностью закрыта от случайных прикосновений специальной шторкой, которая отодвигается только внутри накопителя.

На очереди двухдюймовые дискеты с еще большей емкостью. Первые подобные накопители уже функционируют.

Как же устроен дисковод современного персонального компьютера? Принцип записи, в общем, схож с тем, что используется в обычных магнитофонах. Цифровая информация преобразуется в переменный электрический ток, поступающий на магнитную головку, а затем передается на магнитный диск, но уже в виде магнитного поля, которое диск может воспринять и запомнить. Магнитное покрытие диска представляет собой множество мельчайших областей самопроизвольной (спонтанной) намагниченности. Для наглядности представьте себе, что диск покрыт слоем очень маленьких стрелок от компаса, направленных в разные стороны. Такие частицы-стрелки называются доменами. Под воздействием внешнего магнитного поля собственные магнитные поля доменов ориентируются в соответствии с его направлением. После снятия действия внешнего поля на поверхности диска образуются зоны остаточной намагниченности. Таким образом сохраняется записанная на диск информация. Участки остаточной намагниченности, оказавшись при вращении диска напротив зазора магнитной головки, возбуждают в ней электродвижущую силу, изменяющуюся в зависимости от величины намагниченности.

Как уже упоминалось, информация на диске организована в виде концентрических колец-дорожек (в некоторых компьютерах применяется спиральная дорожка). Дорожки разделены на отдельные участки — сектора. Начало каждого сектора определяется специальной записью, которая производится в процессе форматирования (разметки) диска. Начало нулевого сектора отмечено небольшим отверстием недалеко от центра диска (на некоторых дисках подобным образом отмечено начало каждого сектора).

Накопители, предназначенные для машин класса IBM PC XT и IBM PC AT, обеспечивают 40- или 80-



дорожечную запись информации. Диск с 40 дорожками может иметь емкость 320 Кбайт (в старых машинах типа IBM PC поддерживалось 8 секторов на дорожке) или 360 Кбайт (9 секторов на дорожке для машин IBM PC XT). Машина IBM PC AT оснащается 5.25-дюймовым накопителем высокой плотности емкостью 1.2 Мбайта. В этом случае диск содержит 80 дорожек по 15 секторов. Все дисководы совместимы сверху вниз — то есть на новых моделях можно читать старые диски.

Диск приводится в движение специальным двигателем, обеспечивающим очень стабильную скорость вращения. Это необходимо для того, чтобы каждый бит информации занимал строго определенное место на дорожке (а их 512x8 = 4096 в секторе, не считая служебных), независимо от того, на каком конкретном дисководе была произведена запись. Скорость вращения составляет для дисковода двойной плотности 300 об/мин, а для дисковода высокой плотности 360 об/мин.

Второй двигатель — шаговый. Он предназначен для позиционирования головки на дорожке. С его помощью головки перемещаются по радиусу от края диска к его центру.

Работой всех узлов дисковода управляет специализированный контроллер, который можно считать неотъемлемой частью накопителя. Он включает и выключает привод диска, задерживает его выключение на несколько секунд для ускорения доступа к данным в случае повторного обращения. Контроллер находит нужную дорожку и устанавливает головку на ее середину. Для этого сначала головка устанавливается в зону нулевой дорожки, и производится считывание записанной здесь информации, содержащей адрес искомой дорожки, после этого шаговый двигатель перемещает туда головку. При необходимости, для достижения наилучших условий считывания или записи информации, положение головки на дорожке уточняется. Несколько миллисекунд контроллер ожидает, когда под головкой пройдет требуемый для работы сектор, затем производится запись информации на диск. При этом контроллер осуществляет преобразование информации из параллельного кода, используемого в компьютере, в последовательный, необходимый для записи (в случае чтения происходит обратное преобразование). Контроллер записывает также дополнительную информацию, необходимую для управления работой дисковода и слежения за состоянием информации, а также проверяет, заклеен ли вырез в пластиковом "конверте" диска. Заклеивая вырез, вы тем самым предохраняете диск от стирания.

И.Вязаничев, И.Липкин

Обучающий курс журнала LAN Magazine представляет собой серию статей по вопросам локальных сетей для начинающих пользователей. В этом курсе в простой и доступной форме излагаются основные концепции, лежащие в основе организации локальных сетей. Каждый месяц в сборнике КомпьютерПресс будет печататься очередной выпуск серии, посвященный какому-либо вопросу, связанному с организацией локальных сетей.
Вырезайте и сохраняйте выпуски серии и вы сможете получить в конце обучающего курса брошюру, которая будет представлять собой введение в локальные сети.
В этом выпуске будут рассматриваться вопросы, связанные с принципами отказоустойчивости локальных сетей.

Локальные сети от А до Я:

курс обучения

ЧАСТЬ 16 ОТКАЗОУСТОЙЧИВОСТЬ ЛОКАЛЬНЫХ СЕТЕЙ

Пожалуй, одной из основных проблем, возникающих при эксплуатации локальных сетей, является обеспечение их отказоустойчивости. Многие фирмы, связанные с разработкой программного и аппаратного обеспечения для локальных сетей, уделяют особое внимание способам защиты систем от таких "неприятностей", как внезапное отключение электропитания, отказ сетевого диска или выход из строя файл-сервера.

Традиционными методами защиты компьютерных систем, в том числе и локальных сетей, от отказов, являются процедуры копирования и восстановления данных. В некоторых случаях достаточно продуманная и постоянно осуществляемая схема архивирования файлов на различные магнитные носители (стример, съемный жесткий диск, флоппи-диски) позволяет обеспечить высокий уровень отказоустойчивости, поскольку при возникновении в системе какого-либо сбоя, приведшего к потере информации, достаточно выполнить процедуру восстановления данных из архива. В последнее время появилось достаточно много специализированных программ копирования и восстановления данных, которые работают в среде локальных сетей. В числе наиболее эффективных можно указать Fastback Plus фирмы Fifth Generation, Back-It фирмы Gazelle Systems, PC Tools Deluxe фирмы Central Point Software, BackEZ фирмы EZ-Logic, Corefast фирмы CORE и некоторые другие. Эти программы позволяют выполнять такие процедуры, как копирование всего сетевого диска, копирование файлов рабочей станции на файл-сервер, а также копирование каталогов пользователей с файл-сервера на любую рабочую станцию. Кроме того, практически все программы копирования и восстановления используют алгоритмы сжатия данных и коррекции ошибок, что облегчает процедуру архивирования (уменьшает количество используемых магнитных лент или флоппи-дисков) и повышает ее надежность.

В качестве иного подхода к обеспечению отказоустойчивости локальных сетей следует отметить использование систем с аппаратной избыточностью. Программная поддержка в данном случае осуществляется на уровне сетевой операционной системы, а сетевая архитектура строится таким образом, что позволяет хранить две идентичные копии данных и программ, благодаря так называемым

"зеркальным" дискам или файл-серверам. "Зеркальные" диски или файл-серверы всегда работают как бы в тандеме. Они одинаково сформатированы, содержат одинаковые прикладные программы и данные.

Отказоустойчивость на уровне операционной системы обеспечивает более высокую степень безопасности данных, чем процедуры копирования и восстановления, поскольку последние поддерживают не динамическое, а статическое архивирование данных, хранящихся на сетевых дисках. Именно по этой причине основное внимание мы уделим отказоустойчивым системам с аппаратной избыточностью.

Идентичные запоминающие устройства

Как уже было сказано, отказоустойчивые системы с аппаратной избыточностью предотвращают потерю информации или выход локальной сети из строя благодаря двум идентичным, динамически поддерживаемым копиям сетевых данных на "зеркальных" дисках или файл-серверах. В таких системах отказ одного "зеркального" диска или файл-сервера не приводит к катастрофическим последствиям, поскольку дублирующее или "вторичное" устройство находится в горячем резерве, т.е. работает одновременно с "основным" устройством и, в случае его отказа, начинает выполнять ту операцию, которая выполнялась "основным" устройством. При этом конечный пользователь даже не подозревает, что в системе произошел какой-либо сбой.

В системе с "зеркальным" отображением дисков сетевой файл-сервер включает в себя так называемую "теневую" пару, которая состоит из первичного и вторичного диска. Оба диска управляются одним контроллером, в связи с чем при записи данных на первичный диск они попадают и на вторичный диск. При отказе одного из жестких дисков работа сети не прервется, т.к. сетевая операционная система автоматически перейдет на работу с исправным устройством.

Существует несколько сетевых операционных систем, позволяющих выполнять настройку отказоустойчивых систем с "зеркальным" отображением дисков. Так, система отказоустойчивости SFT NetWare фирмы Novell позволяет администратору сети назначить дублирование каталогов и таблиц размещения файлов FAT с выполнением проверки данных после их записи. Более того, система SFT NetWare определяет "плохую" область диска и помечает ее соответствующим образом, а затем восстанавливает данные, находящиеся на этом месте, путем их копирования с вторичного диска.

"Зеркальное" отображение дисков в системе SFT NetWare.

"Зеркальное" отображение файл-серверов

Подобно системе с "зеркальным" отображением дисков, система с "зеркальным" отображением файл-серверов состоит из двух, параллельно работающих файл-серверов — первичного и вторичного. Связь между "зеркальными" файл-серверами, как правило, осуществляется с помощью особого кабеля и специализированных интерфейсных адаптеров, которые подключаются к внутренней шине каждого файл-сервера. Такая связь может быть осуществлена через параллельный, последовательный (RS-232) или SCSI интерфейсы.

Файл-серверы "теневой" пары постоянно контролируют работу друг друга, поэтому при отказе первичного сервера, вторичный автоматически принимает на себя управление сетью. Такое переключение файл-серверов не отражается на работе пользователей сети, поскольку жесткие диски вторичного сервера содержат зеркальные копии жестких дисков первичного.

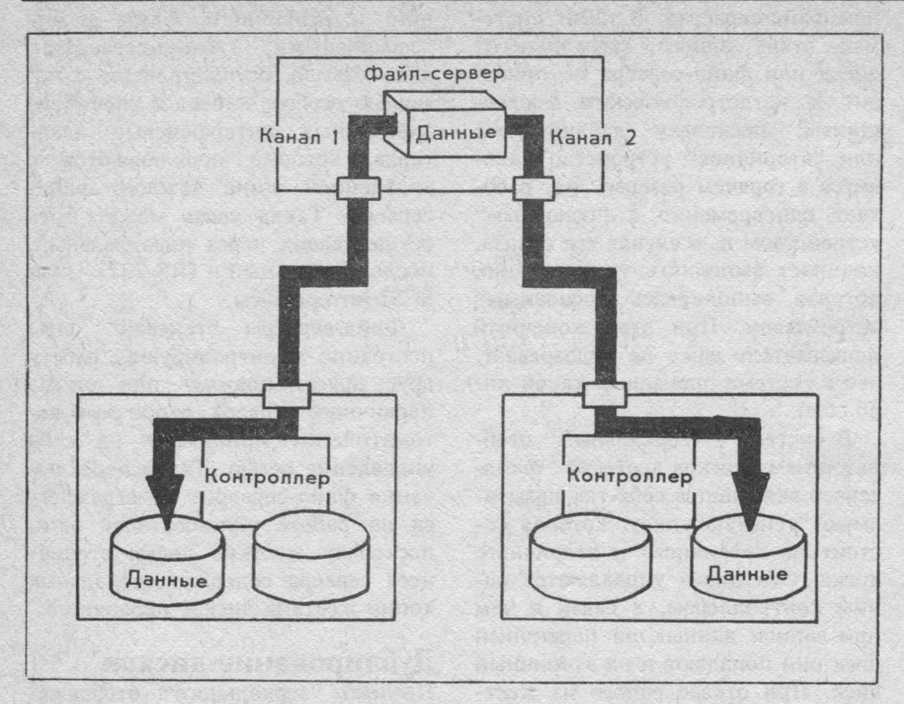
Дублирование дисков

Помимо "зеркального" отображения дисков или файл-серверов в некоторых операционных системах предусмотрен механизм дублирования дисков. В отличие от "зеркального" отображения, при дублировании используется не один, а два дисковых контроллера, что позволяет иметь два независимых дисковых канала. Таким образом, даже при выходе из строя одного из дисковых контроллеров либо интерфейсов вся система останется в работоспособном состоянии.

При дублировании дисков процессор файл-сервера будет считывать данные с того дискового канала, который сработает первым. Следует отметить, что дублирование позволяет повысить производительность локальной сети, т.к. большинство запросов связаны с чтением данных. В связи с этим, файл-сервер может считывать данные с одного диска, записывать на второй диск, а затем, по завершении процесса ввода-вывода, создать "зеркальную" копию записанных данных.

Отслеживание транзакций

Системы отслеживания транзакций представляют собой еще одну форму обеспечения отказоустойчивости



Дублирование дисков в системе SFT NetWare

сети. Она позволяет предотвратить потерю пользовательских данных в случае прерывания не завершенной транцакции. В некоторых сетевых операционных системах подобный механизм является основным элементом обеспечения отказоустойчивости.

Типичная система отслеживания транзакций рассматривает действия с файлами данных, как набор транзакций, каждая из которых должна быть полностью выполнена. При выполнении определенной транзакции, связанной с корректировкой данных на диске, запрещается произведение каких-либо других промежуточных записей на диск. Если во время выполнения такой транзакции происходит сбой, система прекращает выполнение этой транзакции и восстанавливает первоначальное состояние файла данных, после чего транзакция может быть запущена повторно.

В некоторых случаях система отслеживания транзакций используется совместно с администратором записей типа Btrieve фирмы Novell. Это особенно полезно при разработке отказоустойчивых систем управления базами данных, поскольку позволяет поддерживать "прозрачные" связи между прикладными задачами и системой отслеживания транзакций.

Стоит ли тратить деньги?

Один из ключевых вопросов, связанных с использованием отказоустойчивых систем в локальных сетях, звучит примерно так: "Оправданы ли затраты на использование дополнительного оборудования системы отказоустойчивости?". Ответить на этот вопрос не всегда просто, поскольку затраты на систему отказоустойчивости могут быть достаточно большими, если учесть тот факт, что приходится платить за два наиболее существенных компонента локальной сети: файл-сервер (при "зеркальном" отображении файл-серверов) или жесткие диски (при "зеркальном" отображении или дублировании жестких дисков). Чтобы не осталось никаких сомнений, что отказоустойчивость — дорогое удовольствие, к этому следует добавить стоимость соответствующего программного обеспечения.

Для оценки необходимых затрат прежде всего следует определить "степень риска", связанного с потенциальными потерями информации, простоем сети и ее восстановлением при возникновении сбоя. Провести такую оценку достаточно сложно, поэтому многие фирмы-поставщики систем отказоустойчивости предоставляют специальные таблицы и формулы, которые помогают определить "степень риска", спектр оборудования и программ для обеспечения соответствующего уровня надежности.

В конечном счете, многие администраторы локальных сетей сходятся во мнении, что обеспечение высокого уровня надежности и безопасности всей системы стоит своих затрат.

В.Миропольский, В.Демидов

По материалам:
"The LAN tutorial series", LAN Magazine, January 1990.
"Ratings Report", Software Digest, March 1990.



В *октябре 1990 года в Выставочном центре на Красной Пресне в Москве прошла международная выставка "Информатика-90". Многие советские и зарубежные фирмы в области вычислительной техники и программного обеспечения представили на ней свои последние разработки. Представители некоторых из них любезно согласились ответить на вопросы корреспондентов КомпьютерПресс.*

ИНФОРМАТИКА-90

Г-н Эрнст Раушер, специалист по техническим вопросам и лицензированию фирмы Mannesmann Tally. Фирма специализируется по выпуску различных принтеров и другого периферийного оборудования.

КомпьютерПресс: Каковы Ваши концепции бизнеса в СССР, есть ли у вас партнеры?

Эрнст Раушер: Мы только начинаем нашу деловую деятельность на вашем рынке и, конечно, у нас еще нет партнеров. У нас есть контакты с советской организацией в г. Смоленске, которая называется "Искра". Возможно, в будущем эта организация станет нашим деловым партнером, и мы думаем продать лицензию на наши изделия.

КП: Вы собираетесь торговать на рубли или на валюту?

Э.Р.: Только на валюту.

КП: Какие новые изделия вашей фирмы представлены на этой выставке?

Э.Р.: Например, это печатающие устройства типа МТ 130-131, 9-ти игольчатые. В это семейство печатающих устройств также входят 24-х игольчатые печатающие устройства. Печатающее устройство типа 130 на 80 колонок, а типа 131 — на 136 колонок. Быстродействие печатающих устройств, соответственно, 300 и 150 символов в секунду.

КП: Вы продолжаете производить лазерные печатающие устройства?

Э.Р.: Да, мы производим лазерные, струйные и матричные принтеры, а также построчные печатающие устройства, т.е. весь спектр печатающих устройств.

КП: Представлены ли какие-либо лазерные печатающие устройства на этой выставке?

Э.Р.: На этой выставке представлена одна модель лазерных печатающих устройств МТ905. Быстродействие этого принтера — 6 страниц в минуту. Он совместим с эмулятором ТХБ серии 2. Другая модель — МТ906 — тоже обладает быстродействием 6 страниц в минуту. Самая большая модель — это МТ910. Быстродействие этого принтера составляет 10 страниц в минуту. Он имеет две кассеты и стандартную эмуляцию.

КП: Вы впервые участвуете в такой выставке в СССР?

Э.Р.: Да, впервые.

КП: Как Вы считаете, какие перспективы у Вашей продукции на советском рынке?

Э.Р.: Я считаю, что в будущем нас ожидают большие перспективы. И если дела пойдут, то это принесет выгоду обеим сторонам.

КП: Предполагает ли фирма создать совместное предприятие?

Э.Р.: У нас есть кое-какие идеи насчет создания совместного предприятия. Вполне вероятно, что в будущем мы пойдем на это.

КП: Как фирма предполагает преодолеть проблему продажи в СССР за советские рубли? Это важный вопрос для советских покупателей.

Э.Р.: В настоящий момент продажа за рубли невозможна.

КП: В таком случае, что Вы ожидаете от этой выставки?

Э.Р.: Мы привезли наши изделия для продажи и надеемся завязать контакты.

КП: У Вас уже намечены какие-либо советские организации или есть предварительные договоренности?

Э.Р.: Как я уже сказал, мы имеем договоренность с фирмой "Искра" на продажу советской стороне 3000 печатающих устройств. Это наша первая сделка.

КП: Как насчет бартера?

Э.Р.: Торговым представителем является г-н Вортценмайер. Он подойдет позже.

Г-н Маркку Мартинмяки, представитель фирмы Nokia Data. Фирма выпускает мини-ЭВМ, персональные компьютеры, сетевое оборудование и программное обеспечение.

КомпьютерПресс: Какие новые изделия Вашей фирмы представлены на этой выставке?

Маркку Мартинмяки: У нас представлены компьютеры на базе 386-х микропроцессоров, а нашим новым продуктом является оборудование для вычислительных сетей.

КП: Вы впервые участвуете в такой выставке в СССР или Вы являетесь их постоянным участником?

М.М.: Как правило, в Москве мы принимаем участие в выставках раз в год. Начало было положено в 1976 году, и с того времени мы принимаем участие в одной или двух выставках ежегодно.

КП: Вы уже участвовали в других выставках в этом году?

М.М.: Нет.

КП: Считаете ли Вы, что компьютерный рынок перспективен для Вашей фирмы?

М.М.: Конечно.

КП: А в чем это выражается?

М.М.: Прежде всего мы продаем не только компьютеры, но и системы. В настоящий момент у советских покупателей нет денег. Я имею в виду валюту и конвертируемые рубли. Но в то же время существует много организаций, которым требуется огромное количество компьютеров. Но поскольку эти организации не обладают большими денежными ресурсами, у них есть тенденция к покупке дешевых компьютеров. Наша задача заключается в продаже систем, работающих в реальном масштабе времени, которые могут обеспечивать связь Москвы с Ленинградом или с любым другим городом, и спрос на такие системы растет, хотя и медленно. Те организации в СССР, которым необходимы такие высококачественные и надежные системы, и являются нашими заказчиками, хотя их количество растет довольно медленно.

КП: Предполагаете ли Вы в этой связи распространять в СССР сетевое оборудование?

М.М.: Мы его уже поставляем для локальных сетей, а также для широкомасштабных сетей. У нас есть системы, которые обеспечивают связь между городами. Эти системы работают в реальном масштабе времени.

КП: Каковы ваши взгляды на то, каким образом можно преодолеть проблемы оплаты?

М.М.: Я полагаю, вы знаете, что система клиринга между Финляндией и СССР существует недолго. Но пока она существует, мы будем использовать ее возможности. Другим решением проблемы является использование твердой валюты, что мы и делали, но в меньшей степени по сравнению с системой расчетов по клирингу. В некоторых случаях мы готовы пойти на бартерные сделки в том или ином виде, но это не значит, что мы будем покупать то, что нам не нужно. Для бартера необходим третий партнер, который представлял бы какую-либо финскую компанию. Так что мы иногда использовали такую форму бизнеса, и вполне успешно.

КП: В связи с этим такой вопрос: с какими советскими организациями Вы имеете устойчивые связи?

М.М.: Все организации трудно перечислить, но я рискну назвать несколько из них. Это Верховный Совет СССР, Минздрав, АвтоВаз, Внешэкономбанк, Промстройбанк, Жилсоцбанк, Минэнерго, Союзгаз и т.д.

КП: Какое оборудование Вы продаете?

М.М.: Раньше мы продавали мини-ЭВМ, а сегодня мы продаем мини-ЭВМ для локальных сетей, модемы и терминалы.

КП: Могли бы Вы назвать конкретные модели компьютеров?

М.М.: Сегодня мы компонуем системы так, что они включают в себя в качестве серверов связи (шлюзов) и серверов баз данных 386-е компьютеры, а рабочие станции — 286-е компьютеры. Мы тесно сотрудничаем с фирмами Microsoft и Intel, являясь опытным пользователем изделий фирмы Intel в Европе. А также раньше других пользователей получаем от фирмы информацию о ее новых изделиях.

КП: Существуют ли совместные предприятия с Вашим участием и если нет, то намерены ли Вы создать совместные предприятия?

М.М.: Если говорить о концерне Nokia, то существуют два совместных предприятия: одно из них организовано совместно с кабельной промышленностью, другое — с министерством связи. Но Nokia Data не участвует в совместных предприятиях. Наше решение, хотя мы и ведем переговоры, не совместное предприятие, потому что у нас есть другие пути для реализации наших конкретных задач.

КП: Некоторые сейчас считают, что СП не так эффективны, как хотелось бы. В связи с этим, каковы перспективы продажи за рубли?

М.М.: Это основная причина, по которой мы не стали организовывать СП. Они не решают всех проблем. У нас есть конкретные задачи, которые мы можем решить и без СП. Я слышал, что наше отделение, занимающееся вопросами связи, столкнулось с проблемой, как использовать свое совместное предпри-

ятие. Но все же их совместные предприятия приносят прибыль, потому что они входят в немногочисленную группу совместных предприятий, которые занимаются настоящим делом.

КП: Что Вы ожидаете от этой выставки?

М.М.: Доработать контракты, которые уже есть, и заключить новые. Вы знаете, что в СССР это вопрос не одной недели. Это может занять год или два. Мы вчера заключили небольшой контракт и ожидаем, что подпишем еще один во вторник. Но эти контракты не являются результатом выставки. Они являются результатом долгой кропотливой работы. На выставке, в большинстве случаев, мы просто демонстрируем изделия и их возможности.

КП: Небольшой вопрос об объемах производства, объемах продаж?

М.М.: Оборот Nokia Group — на уровне 28 млн. финских марок, т.е. приблизительно 7 млн. долл. На Nokia Data приходится 5 млн. финских марок, или около 1 млн. долл. Наша доля в СССР очень маленькая по двум причинам: первое — это эмбарго, хотя в настоящий момент происходят значительные перемены в лучшую сторону. Второе — наше оборудование не является самым дешевым на советском рынке. У нас особые заказчики, которым требуются системы высокого качества, которые включают наши компьютеры и модемы.

Г-н Алан Партридж, коммерческий представитель компании Quest Automation. Фирма продает аппаратное и сетевое обеспечение, а также системы САПР.

КомпьютерПресс: Первый вопрос, естественно, о тех новых изделиях, которые представлены Вами на выставке.

Алан Партридж: Все меняется. Меняются и нормы КОКОМа, поэтому мы сейчас можем продавать машины на базе 386-х и 486-х микропроцессоров. На стенде представлен наиболее современный сервер файлов Apricot на базе 486-го микропроцессора. Эта машина пользуется здесь большим интересом, поскольку многие хотят установить сети, а файл-сервер является как бы ядром сети. Он и является нашим главным экспонатом. Кроме того, мы недавно стали основным дилером PCAD, т.е. системы автоматизированного проектирования печатных плат. Пакет PCAD широко известен в СССР, но до сих пор официально не продавался в вашей стране, поэтому мы надеемся, что советские заказчики проявят интерес и закупят последнюю версию этой системы. AUTOCAD также является основным изделием, программным продуктом, которым мы торгуем. Традиционно наш бизнес в СССР основывался на программных системах и САПР для проектирования печатных плат, но с приходом персональных компьютеров мы смогли продавать прикладное программное обеспечение более широкого профиля для систем САПР. Сейчас мы предлагаем более широкую гамму персональных компьютеров, чем ранее. Мы продаем персональные компьютеры фирм IBM, Compac, Apricotr, а также целый ряд клонов, программных продуктов, а также и те программные продукты для персональных компьютеров, которые есть на Западе.

КП: Насколько мы знаем, одной из перспектив на советском рынке являются локальные сети. В связи с этим хотелось бы узнать, какие коммерческие перспективы на советском компьютерном рынке?

А.П.: Как я уже сказал, мы продаем сети фирмы Novell, которые в СССР, в отличие от Запада, еще не стали промышленным стандартом. Но те, кто интересуются сетями, автоматически подразумевают сети Novell. Существуют два протокола ArcNet и Ethernet, а выбор той или иной системы зависит от расстояния и количества терминалов в сети. Но мы являемся организацией, которая торгует изделиями, право на продажу которых принадлежит одной фирме. Мы не переделываем и не разрабатываем системы и программное обеспечение, которыми торгуем. Но это не означает, что мы не будем поставлять то лучшее, что может появиться. Однако, в настоящее время системы сетей Novel являются самыми современными.

КП: Вы собираетесь продавать русифицированные версии таких систем, как PCA и прочее?

А.П.: Уже существует русская версия AutoCAD, но вряд ли будет русская версия PICA, короче говоря, все будет зависеть от объема продаж и потенциальных продаж этого изделия. Мы обнаружили, что советские пользователи, на примере последней русифицированной версии AutoCAD, предпочитают использовать английскую версию, потому что они уже привыкли к меню и т.д. Другая причина заключается в том, что при русификации базового AutoCAD многие программы, которые стыкуются с системой СА, не стыкуются с русскими версиями. Поэтому применение русской версии более ограничено, чем применение английской версии. Например, одно из дополнительных программных средств AutoCAD, которое представляет особый интерес в СССР, это пакет для архитектурного проектирования, но он не стыкуется с русской версией. Так что насчет русификации таких стандартных пакетов, как AutoCAD и PCAD существуют сомнения. При необходимости те, кто поставляют такие системы, всегда готовы рассмотреть этот вопрос. Еще одной областью, которая представляет интерес для советских покупателей, является наше программное обеспечение для баз данных с установкой сетей. Обычно, заказчики говорят: “Хорошо, мы знаем и имеем сеть, но у нас нет базы данных“.

КП: В этой связи какие базы данных Вы предполагаете продавать на советском рынке?

А.П.: Мы продаем такие продукты как Framework 3, dBASE для баз данных и Article.

КП: Это интересно, так как dBASE очень распространен в СССР.

А.П.: Article относительно новое программное средство для СССР, потому что все зависит от приложения. У нас есть заказчик в области медицины, который хочет включить базу данных в сеть, и этот заказчик запросил наши рекомендации. Мы порекомендова-

ли Article, поскольку она является довольно всеобъемлющей базой данных. С другой стороны, Article может оказаться очень большой системой, а dBASE 3 или Framework 3 как раз то, что надо для заказчика.

КП: Каким образом Ваша фирма предполагает решить проблему конвертируемой валюты?

А.П.: Это та проблема, которую мы пытаемся преодолеть каждый день. С той системой, которая существует в СССР, трудно вести дела “в рублях“. Прежде всего мы должны платить в твердой валюте, хотя наша фирма и аккредитована в СССР 20 лет назад, за те удобства, которые нам предоставляют. Так что сама система не дает нам возможности работать за рубли. Предположим, у нас есть рубли, но ведь мы не можем их использовать. У нас есть один солидный заказчик, с которым мы обсуждали вопрос конвертируемости рубля, но опять же все зависит от 100 дней плана в 500 дней. Но никто не знает, пройдет ли год или два.

КП: Как вы относитесь к бартеру?

А.П.: У нас есть много таких предложений, но мы никогда не имели выгоды от бартерных сделок, в основном потому что у нашей организации нет такой структуры, которая могла бы использовать бартерные сделки. Мы заключили ряд контрактов на индийские рупии и мы знаем, что СССР располагает значительной суммой египетских фунтов и мы пытаемся решить непростую проблему египетских фунтов. Таким образом, кроме твердой валюты, существуют и другие деньги, такие как индийские рупии, финские марки, сирийские фунты. И если бы мы могли найти пути для торговли на эти виды валют, мы бы пошли на это. Хорошо бы найти способ перевода рублей в доллары!

КП: Не считаете ли Вы, что совместные предприятия являются таким выходом?

А.П.: У нас уже есть совместное предприятие под названием “Трио“, которое действует почти в течение полутора лет.

КП: Его деятельность эффективна?

А.П.: С точки зрения маркетинга СП кое-что сделало, но в целом оно не так эффективно, как нам хотелось бы. Однако, мы открыли на СП производственные помещения, так что все работы по конфигурированию будут выполняться в СССР. Обычно такая работа выполнялась в Великобритании. Это один из подходов к решению проблем, связанных с заказами. Например, у нас был заказ на систему, включающую в себя определенное число периферийных устройств. Мы составили ее конфигурацию, провели проверку перед отгрузкой покупателю. Раньше все это делалось в Великобритании. В общих чертах, СП дает минимальный экономический эффект, но зато СП обеспечивает базу для заключения “рублевых“ контрактов.

КП: Как Вы считаете, есть ли еще какие-либо формы бизнеса?

А.П.: Мы обсуждали вопрос о передаче функций оптовой торговой фирмы одному нашему крупному заказчику, поскольку он представляет большое предприятие и связан с другими предприятиями. Иногда встает вопрос о продаже технологий предыдущих уровней. Например, если бы мы могли получить много компьютеров типа XT, то мы смогли бы продавать такую большую партию за рубли, но не напрямую, а через оптовых торговцев.

КП: С кем из советских организаций у Вас уже установились связи?

А.П.: Поскольку наша фирма работает в СССР почти 20 лет, то у нас большое количество клиентов. Немного найдется советских организаций, которые не знают о нашей фирме. Ранее мы продавали системы автоматизированного проектирования и изготовления печатных плат и фотоплоттеров. По Союзу продано и установлено около 700 или 800 фотоплоттеров. У нас широкие контакты с советскими организациями. Я бы хотел упомянуть еще одну область, которой мы заинтересовались. В СССР много машин местного производства.

Я не могу точно сказать где, но мы недавно установили подновленную систему IBM 4341 вместе с НМД и НМЛ, т.е. полную конфигурацию. Мы надеемся на расширение такого рода деятельности, т.е. поставку обновленного оборудования фирмы IBM пользователям советских машин, независимо от серии машин IBM (серии 4300 или 9700) с родной периферией, т.к. на Западе компании меняют компьютеры и технологии, которые слегка устарели там, но все еще используются в СССР. И такие технологии продаются по доступным ценам. При первой установке упомянутой системы мы обнаружили, что можно установить систему IBM и подключить периферийные устройства IBM к советским машинам или советскую периферию к машине IBM. Мы еще не проверили, можно ли подключить процессоры IBM к процессорам советских машин различного ряда. Но фактически мы можем установить машину IBM в одном помещении с советской машиной и коллективно пользоваться периферийными устройствами независимо от того, кто является их производителем. В этом направлении мы и надеемся работать.

КП: Что Вы ожидаете от этой выставки?

А.П.: Мы надеемся продолжить нашу работу в СССР. Мы считаем, что западные компании переживают не самое лучшее время в СССР. Надеемся, что ситуация изменится к лучшему, а сейчас нам надлежит приложить больше усилий для успешной деятельности, для поддержания того уровня бизнеса, который бы оправдывал наше пребывание в СССР. Наши планы связаны с обеспечением конкурентоспособного изделия, чтобы заполучить те сделки, которые доступны в данный момент, а также чтобы продлить свое пребывание в СССР и воспользоваться теми преимуществами, которые появятся при улучшении ситуации.

Г-н Норберт Браун, коммерческий директор по экспорту на Восточную Европу фирмы Epson.

КомпьютерПресс: Первый вопрос, который мы бы хотели Вам задать таков: какие новые изделия Вашей фирмы представлены на стенде?

Норберт Браун: Две новых модели персональных компьютеров и четыре новых печатающих устройства.

Это оборудование выставляется впервые не только в СССР, но и в Восточной Европе.

КП: Вы впервые участвуете в выставке в СССР?

Н.Б.: Нет. У нас сложились традиционные деловые отношения с СССР в течение уже многих лет. Что же касается печатающих устройств, то фирма Epson является мировым лидером на рынке. То же относится и к Советскому Союзу. Мы работаем в Союзе уже в течение 6-7 лет, и если информация, которой я располагаю, правильна, то нам принадлежит более 50% всего рынка импорта печатающих устройств.

КП: Считаете ли Вы компьютерный рынок в СССР перспективным для Вашей фирмы?

Н.Б.: Что касается перспективы в будущем, то трудно сказать, будет ли у нас рынок, сравнимый с тем рынком, который мы имеем сейчас в США. Следует также различать спрос и реальную покупательную способность. Спрос огромный, но это не означает, что бизнес процветает. Так что мы терпеливо следим за развитием событий, как мы уже это делали в прошлом. Мы стараемся показать себя хорошими партнерами советских клиентов независимо от того, являются ли клиенты представителями внешнеторговых объединений или это вновь образованные ассоциации, с новыми правами СП, кооперативы и т.д., потому что считаем, что рыночная система изменяется и будет изменяться.

Но трудно сказать, когда ситуация изменится. Мы понимаем те трудности, с которыми столкнулся Советский Союз, начиная с этого года. Нехватка твердой валюты, а также все остальные известные вам проблемы, которые не имеет смысла перечислять. Но мы все равно участвуем в выставках и не только в Москве. В прошлом году мы участвовали в выставке в Киеве. Мы также были готовы к выставке в Киеве и в этом году, но к нашему удивлению, та выставка была как бы одноразовая и в этом году не состоялась. Мы полностью за выставки, но при этом должна быть какая-то преемственность. Иначе мы не знаем, будет ли организована выставка на следующий год. Выставки должны быть традиционными, постоянными.

КП: В связи с нехваткой валюты какие пути Вы видите для преодоления этой проблемы?

Н.Б.: Если бы у меня было готовое решение, то я бы поделился им с г-ном Горбачевым. Но у меня нет такого решения. Я думаю, что эту проблему должен решать сам Советский Союз и что иностранные компании только до определенной степени должны участвовать в решении этого вопроса путем поддерживания деловых отношений с СССР. Но деловые отношения базируются на рынке, а положение в СССР весьма сложное. С нашей стороны мы можем предложить продолжение деловых отношений, проявление заинтересованности в деловых операциях с СССР. Вот поэтому-то мы и участвуем в этой выставке. Не потому что мы ожидаем крупных сделок, а чтобы подтвердить нашу заинтересованность в вашей стране. Поэтому мы продолжаем предлагать свои услуги.

КП: Вы упомянули о сложной экономической ситуации, тем не менее имеют ли фирмы устойчивые торговые контакты с определенными советскими торговыми организациями?

Н.Б.: Что касается СССР, то мы продолжаем вести деловые отношения непосредственно по нескольким каналам. Одним из каналов является прямая продажа наших изделий непосредственно нашим клиентам.

КП: И последний вопрос. Что Вы ожидаете от этой выставки?

Н.Б.: Прежде всего ожидаем, что мы “прочувствуем“ настоящее положение, чтобы решить, какими путями и способами мы можем помочь нашим традиционным заказчикам, новым заказчикам подготовить себя к тому времени, когда экономическая ситуация в СССР снова нормализуется. Я жду этого и надеюсь, что положение изменится начиная со следующего года. Все зависит от времени и от внешнеторгового баланса... Прошлый год, с точки зрения дебита, для СССР был ужасным. Надеюсь, что меры, которые будут приняты советским правительством, приведут к кредитным соглашениям в конце этого года и тогда положение снова нормализуется. Благодаря огромному спросу, модернизация промышленности будет продолжаться, а при модернизации промышленности необходимо иметь оборудование для обработки данных. Без такого оборудования невозможно профессионально модернизировать промышленность, достичь определенного стандарта, чтобы конкурировать с западными производителями в западных странах. Это и есть та проблема, которую надлежит решить, а также цель каждой производящей и экспортирующей отрасли промышленности.

Г-н Антон Польстерер, глава представительства фирмы Hewlett-Packard.

КомпьютерПресс: Какие новые изделия фирмы представлены на выставке?

Антон Польстерер: У нас столько новых изделий, что, к сожалению, на нашем стенде не хватит места. Вы, вероятно, знаете, что в июле этого года был изменен перечень оборудования, которое мы можем передавать в СССР. Сейчас уже мы можем передавать в СССР большинство из того оборудования, которое производится фирмой. Например, мы привезли целую серию рабочих станций для широкого применения в различных областях, таких как электронное конструирование, несколько пакетов для применения в области машиностроения при участии таких партнеров, как Delta, McDonald Douglas, а также наши собственные программные пакеты выполнения проектных работ в области машиностроения. Еще мы привезли на выставку многопользовательскую вычислительную систему для использования в производстве, т.е. систему комплексного интегрированного производства (CIM). Это и предлагаемые различные программные пакеты для выполнения различных производственных заданий.

КП: Мы знаем, что Hewlett-Packard в основном производит оборудование. Вы упомянули о том, что

представляете не только оборудование, но и программное обеспечение. Это программное обеспечение фирмы Hewlett-Packard или Вы представляете программное обеспечение, разработанное вашими партнерами?

А.П.: Хороший вопрос. Один из наших лозунгов на этой выставке следующий: "Мы представляем программные решения наших партнеров по всему миру". Так что здесь мы представляем программные пакеты таких наших партнеров, как Delta, McDonald Douglas плюс наши собственные пакеты, такие как программные пакеты для применения в области машиностроения, пакеты для управления производством и планирования производства, финансового учета. Мы представляем сочетание программных пакетов. Упор делается не только на продажу аппаратных средств, но и программных пакетов. Что касается рабочих станций и систем, то они занимают почти половину стенда. Не следует также забывать о такой важной области, как периферийные устройства и персональные компьютеры, но они на другой стороне стенда, а также на контрольно-измерительные приборы. Это особенно относится к тем приборам, которые используются в компьютерной индустрии, такие как анализаторы цепей, спектра, логические анализаторы.

КП: Значит ли это, что политика фирмы Hewlett-Packard в настоящий момент заключается в том, чтобы поставлять продукцию "под ключ в комплексе", т.е. и аппаратное и программное обеспечение?

А.П.: В определенных областях. Если взять, например, машиностроение, то мы поставляем оборудование и программный пакет. Если взять другую область, такую как автоматизированное проектирование в сфере производства электроники, то у нас есть партнер, который поставляет и аппаратное и программное обеспечение. Это фирма Deltagraphics. В области управления производством мы поставляем оборудование и программное обеспечение. Другие партнеры поставляют для производства наше оборудование и их программное обеспечение. Но мы не интегрируем систем. Есть специальные фирмы, которые занимаются интегрированием систем "под ключ". Мы поставляем компьютерное оборудование и программное обеспечение для определенных областей применения.

КП: Мы знаем, что фирма долгие годы присутствует на советском рынке, предполагает ли фирма продавать советским пользователям русифицированные продукты?

А.П.: Да, мы уже осуществили адаптацию наших персональных компьютеров, а также периферии к персональным компьютерам. Сейчас мы работаем над адаптацией наших программных пакетов, особенно в тех областях, которые имеют особое значение. Так что, мы планируем адаптацию наших программных пакетов, как это делается в Германии. Но существуют пакеты, которые используются в английских версиях. Есть французские, немецкие версии и есть версии, адаптированные на советский рынок.

КП: Считаете ли Вы, что советский компьютерный рынок является перспективным для Hewlett-Packard?

А.П.: Перспективы очень хорошие, потому что мы уже создали "плацдарм", чтобы догнать другие страны. У нас широкие планы, но вопрос состоит в том, когда и как. Это зависит от темпа экономических реформ, от того, насколько готовы будут советские организации к использованию информационной технологии. Но информационную технологию нельзя использовать саму по себе. Все зависит от организации предприятия.

КП: Существуют ли у вас устоявшиеся связи с какими-либо советскими организациями?

А.П.: Это проектные институты в области строительства, машиностроения, архитектуры, банки, библиотеки, КомпьютерПресс, с вашего позволения. Мы можем назвать библиотеку имени Ленина, Петровский пассаж, Гипромез, а также медицинские учреждения. Наша первая компьютеризированная система была предназначена для медицинского анализа электрокардиограмм.

КП: Существует ли для Вашей фирмы проблема валюты? Это интересует наших читателей. Это для нас головная боль.

А.П.: Это действительно проблема, потому что количество твердой валюты у тех организаций, которые имеют к ней доступ, не увеличилось. Для нас проблема найти организации, имеющие твердую валюту. Поэтому нам приходится все время искать новые организации, новых покупателей.

КП: Как насчет контрактного программирования?

А.П.: Мы дали задание нашему отделу, который решает проблемы программирования на контрактной основе, наладить контакты с советскими организациями.

КП: Вы нашли какую-либо советскую организацию, занимающуюся вопросами программирования, которая подходит Вам?

А.П.: Это не та область, где можно действовать быстро. Мы должны лучше узнать друг друга, поработать на основе метода "пробы", решить, можем ли мы работать друг с другом, а затем завязать отношения, что может занять от 6 месяцев до 2 лет. Программирование нельзя сравнивать с остальным бизнесом, когда получаешь предложение и осуществляешь сделку на покупку. Для развития такого сотрудничества требуется время, но раз мы начали налаживать такие контакты, это означает, что мы верим в возможности этого рынка. Возможно, скоро мы подпишем контракты, а в случае успеха, мы заключим еще большее их количество.

КП: Произошли ли какие-то изменения на Hewlett-Packard, влились ли какие-то новые фирмы ?

А.П.: Последней фирмой, которая влилась в Hewlett-Packard, была Apollo Computers, ставшая частью HP более года назад и мы считаем, что рабочие станции и компьютеры отделения HP Apollo будут иметь успех в вашей стране. Мы видим, что к этим компьютерам и рабочим станциям проявляется большой интерес.

КП: Какое, по вашему мнению, генеральное направление наступления HP на советский рынок?

А.П.: Стать главным поставщиком на советском рынке.

КП: Или Вы хотите обойти его со всех сторон?

А.П.: Мы хотим охватить все 15 республик.

КП: Мы знаем, что Hewlett-Packard имеет несколько дилеров. Предполагает ли фирма организацию совместных предприятий? В СССР есть такое мнение, что СП представляют некоторое решение проблемы оплаты в рублях, т.е. какое-то упрощение финансовых механизмов. Каковы планы фирмы в этом отношении?

А.П.: Давайте начнем с дилеров. Персональные компьютеры и периферия во всем мире продаются в основном через дилеров. Мы также организовали дилерскую сеть в вашей стране. Для реализации других изделий необходимы партнеры, такие как у нас есть на Западе: McDonald Douglas, Deltagraphics, Hann Engeneering. Такие же партнеры нам нужны в вашей стране. Может быть некоторые наши изделия будут собираться в СССР. Я знаю, что мы должны проявить определенную активность. Я хочу сделать немного наоборот. Сначала хочу определить направление деятельности, а потом определить структуру. Я не придерживаюсь мнения, что необходимо создать СП, а затем определять задачи его деятельности. Я скорее вступил бы в партнерские отношения. Например, партнерское соглашение об адаптации нашего программного продукта. Многие учреждения занимаются адаптацией наших программных продуктов на советском рынке и разрабатывают дополнительные модули, на которые есть спрос в вашей стране. Мы осуществляем такую деятельность в соответствии с соглашениями о сотрудничестве, и если работа по этим соглашениям окажется успешной, мы будем искать другие формы официального сотрудничества. СССР — большая страна, поэтому нам понадобится множество партнеров. В таком же стиле мы работаем и в других странах и работаем довольно успешно.

КП: Кто на Ваш взгляд является Вашим основным конкурентом на советском рынке?

А.П.: Это не секрет, но все зависит от области бизнеса. Вы знаете, что у нас на фирме 6 отделений: аналитика, контрольно-измерительные приборы, медицина, компьютерные системы и рабочие станции, и последнее, но не менее важное, периферийные устройства. Большинство наших конкурентов работают в области персональных компьютеров. Но конкуренты в этой области торгуют и персональными компьютерами младших моделей, т.е. дешевыми и наименее производительными персональными ЭВМ. Что касается наиболее производительных машин, то большой конкуренции мы не видим. Если вы посмотрите выставку, то вряд ли найдете конкурента по рабочим станциям, многопользовательским системам, как система 3000. В основном вы увидите производителей персональных компьютеров и их клонов. Я рассматриваю персональный компьютер как персональную рабочую станцию того, кто занимается сбором и обработкой информации. Рабочие станции как бы являются окном в мир информации, но еще предстоит организовать “мир информации“ за рабочей станцией. Можно создать информационную сеть на базе персональных компьютеров, чтобы связать различные подразделения на заводе или в организации.

КП: Можно сказать, что основная область Ваших продаж — персональные компьютеры и периферийные устройства?

А.П.: Да.

КП: Считает ли фирма, что в СССР в ближайшем будущем необходимо будет приложить определенные усилия для объединения этих компьютеров и периферийных устройств в локальные сети, и предполагаете ли Вы что-нибудь в этом плане?

А.П.: Что касается персональных компьютеров, то многие наши конкуренты в области персональных компьютеров используют нашу периферию. Многие наши дилеры стыкуют нашу периферию с другими компьютерами. Что касается локальных сетей на базе ПЭВМ, то наши дилеры продают сетевое оборудование. У нас есть сетевое оборудование. Hewlett-Packard является основным производителем широкомасштабных сетей. У нас у самих установлена одна из самых крупных сетей. На фирме работает 90 тыс. человек. Это около 65 тыс. персональных компьютеров и терминалов, объединенных в широкомасштабную сеть.

КП: Это существующая сеть?

А.П.: Да. Первая частная сеть в Европе установлена фирмой Hewlett-Packard, и фирма, которая ее купила, пользуется нашим обслуживанием. Это кампания по прокату автомобилей. Это чисто информационная сеть, включающая в себя серверы, т.е. широкомасштабная сеть. Здесь, конечно, сеть потребует большой организации, в которой необходимо подключение различных компьютеров в единую сеть на международном стандарте. Мы говорим о ведущей советской организации, которая занимается сетями.

КП: Что Вы ожидаете от этой выставки ?

А.П.: Я искренне надеюсь, что мы сможем наладить контакты с новыми заказчиками. Как я уже сказал, процесс принятия решений в вашей стране должен быть децентрализован. Вот и установление сети даст нам возможность поддерживать связь со многими заказчиками. Также важно продавать не только оборудование, но и программные решения, аппаратные и программные пакеты. Чтобы продать нужный программный пакет, необходимо иметь связь с заказчиками. Компьютеризация является положительной тенденцией, а для информационной сети важно поставить ту систему, которая будет отвечать требованиям сети.

В.Демидов, В.Миропольский, А.Агафонов

(Окончание следует)



Этот выпуск, "Между прочим...", посвящен неполадкам, появляющимся при возникновении ошибок четности и тому, как превратить старый компьютер IBM PC XT в компьютер, совместимый с IBM PC AT.

Уважаемые читатели, мы ждем ваших писем с вопросами и с описанием методов решения различных проблем, возникающих при работе на персональных компьютерах. Наиболее интересные предложения будут опубликованы в следующих выпусках "Между прочим..."

ОШИБКИ ЧЕТНОСТИ

Случается, что при работе машины неожиданно появляются сообщения вроде "Parity NMI at 9902:0000 — Type (S)hut off NMI, (R)eboot, or any other key to continue". При этом, если сообщение повторяется, адрес может изменяться. О чем говорит такое предупреждение и нужно ли что-то делать в этом случае?

Такое сообщение системы говорит о том, что в оперативной памяти произошла ошибка четности. Бит контроля четности вычисляется перед записью 16-ричного слова в в оперативную память. Затем этот бит записывается в ОЗУ вместе со словом, хранящемся по адресу 9902:0000. При чтении вновь вычисляется значение бита четности и, если вычисленное значение не совпадет с записанным, возникает сообщение об ошибке, т.к. повреждена информация.

Причиной этого может быть неисправная микросхема оперативного запоминающего устройства (ОЗУ). Чтобы убедиться в том, что нет иных причин появления сбоя, выключите ваш компьютер, снимите с него крышку и удостоверьтесь, что каждая из микросхем ОЗУ (или каждый из модулей, если у вас память выполнена на SIMM) правильно вставлена в колодку. Если машина оснащена расширенной памятью, проверьте как установлена соответствующая плата. Затем включите питание. Если ошибка четности появилась вновь при самотестировании — значит пора везти ваш компьютер в ремонт или самостоятельно браться за диагностику компьютера (а для этого существует много подходящих пакетов) и менять неисправную микросхему.

МОДЕРНИЗАЦИЯ КОМПЬЮТЕРА IBM PC XT

Сейчас многие пользователи IBM PC XT стремятся заменить свой устаревший компьютер на более совершенный. Переход на новую технику обходится очень дорого, и мы хотели бы предложить вам более дешевый и простой путь.

Несколько фирм сейчас выпускают ускоряющие платы на базе процессора 80286, предназначенные для увеличения производительности компьютеров класса IBM PC XT до уровня IBM PC AT без замены основной платы компьютера. Две из них мы опишем.

Первая плата — SOTA 286i — производится фирмой SOTA Technology и предназначена для работы с компьютерами на базе процессоров 8088 или 8086. К ней дополнительно предлагается плата расширения памяти Memory/16i, позволяющая нарастить ОЗУ до 8 Мбайт и работать под управлением OS/2.

Вторая плата изготавливается фирмой Orchid Technology. Она называется Tiny-Turbo XTra и работает с компьютерами, построенными на базе процессора Intel 8088.

Каждая из плат рассчитана 12.5 МГц, но для обеспечения совместимости может переключаться на на тактовую частоту 4.77 МГц. Для уменьшения времени ожидания при работе быстрого процессора с ОЗУ старого компьютера (а также, чтобы избежать необходимости установки быстрого ОЗУ, что стоит весьма дорого), обе платы имеют статическое сверхоперативное запоминающее устройство. В результате установки платы, ваш старый XT превращается в современную быструю машину, на которой выполняется почти любое программное обеспечение, разработанное для IBM PC AT.

Каждая из плат-ускорителей стоит менее 450 долларов, а это, согласитесь, куда дешевле нового компьютера. Плата расширения Memory/16i без микросхем памяти продается за 295 долларов, а вариант с двумя мегабайтами ОЗУ — за 895.

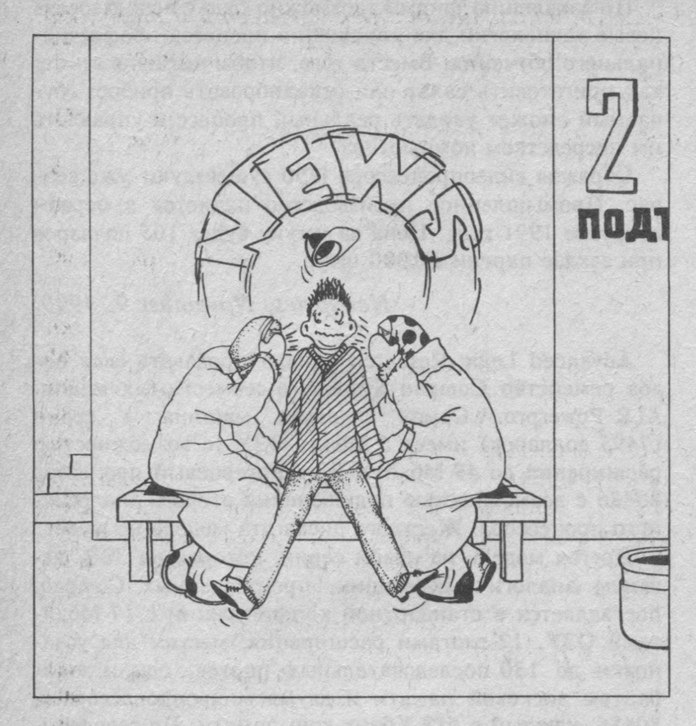
Однако, не забывайте, что шина данных на основной плате остается по прежнему восьмиразрядной и разъемы расширения тоже, а информация по такой шине передается значительно медленнее, чем через по 16-разрядной шине AT. Это особенно существенно при обмене данными с жестким диском.

И.Вязаничев

По материалам:

J.Prosise "Parity Errors", PC/Computing, June 1989.

"Easy Upgrades to AT", PC/Computing, January 1990.



НОВОСТИ

Американская ассоциация прогрессивных связей и Международный фонд за выживание и развитие человечества собираются открыть в Москве еще одну компьютерную сеть — GlasNet (Glasnost Network). По заявлению организаторов, все услуги будут предоставляться за советские рубли. Сеть будет предоставлять возможность обмена электронными сообщениями и файлами с пользователями многочисленных западных систем электронной почты. Предполагаемая оплата за пользование — около 100 рублей в месяц. Сетевой компьютер будет установлен в Москве в конце 1990 года.

Newsbytes, October 20, 1990

В последнее время многие фирмы приступили к выпуску портативных компьютеров, пользующихся на рынке повышенным спросом.

Так, фирма Sony начала производство компьютера QL/Note PCX-310NR, основанного на IBM-совместимой японской архитектуре AX. Фирма заявляет, что ее компьютер лучше других аналогичных машин работает с Microsoft Windows и может присоединяться к локальной сети. Компьютер комплектуется мышью и в различных конфигурациях стоит от 615 до 1692 долларов.

Фирма Toshiba улучшила компьютер DynaBook, оставив прежнюю цену в 1523 доллара. DynaBook286 J-3100GS 001 обрабатывает данные вдвое быстрее предыдущего компьютера, имеет более качественный экран и может работать от батареи 5 часов без перерыва. Компьютер размером с записную книжку поставляется в конфигурации с ОЗУ 1.5 Мбайта с возможностью расширения до 10 Мбайт на дополнительной плате.

Канадская фирма Ogivar начала продажу компьютера Internote 286, весящего без батарей 4.8 фунта (2.16 кг). Он имеет VGA-совместимый плоский экран, встроенную полноразмерную клавиатуру и жесткий диск емкостью 20 Мбайт. Internote продается за 2999 долларов.

Фирма Compaq создала новую технологию "Regal Flex", позволяющую разместить весь компьютер на одной плате. Одна из моделей фирмы, созданная по этой технологии — Compaq LTE 386s/20 — поставляется с 60-Мбайтным диском и модулями расширения памяти размером с кредитную карточку. Ее стоимость — 6999 долларов.

Newsbytes, October 20, 1990

Японская компания Nippon Sheet Glass, специализирующаяся на выпуске прочных сортов стекла, и калифирнийская Areal Technology совместно разработали винчестер, использующий стеклянные диски как носитель иноформации. Стеклянный диск может записывать втрое больше информации, чем нынешние алюминиевые — 60 Мбайт. Это происходит из-за того, что на более гладкой поверхности головка может располагаться ближе к поверхности диска.

Сам стеклянный диск толщиной 1.5 мм, как предполагается, будет пользоваться большим спросом у производителей миниатюрных машин.

Две компании продадут первые опытные образцы дисков в январе по цене 1600 долларов за штуку.

Newsbytes, October 30, 1990

Фирма Hitachi предлагает производителям компьютеров свой новый жесткий диск емкостью 419 Мбайт и диаметром 3.5 дюйма. Повышенная плотность хранения информации и высокоэффективный механизм доступа позволили компании увеличить емкость в 1.7 раза по сравнению с предыдущей моделью.

Привод DK314C-41 имеет стандартный интерфейс SCSI, а время доступа составляет 16.8 мс.

Цена за пробный образец — 3700 долларов. Поставки начнутся в январе 1991 года. За четыре следующих года компания планирует продать до 200 тысяч таких дисков изготовителям рабочих станций и мощных персональных компьютеров.

Newsbytes, November 9, 1990.

Фирма ICM, производящая в основном диски для персонального компьютера PC-9801, начала продажу

винчестеров, установленных на плате контроллера. Устройство, разработанное ICM совместно с Fuji Electric является, по заявлению фирмы, первым продуктом такого рода, имеющим диск емкостью 40 Мбайт и диаметром 2.5 дюйма.

Винчестер на плате сможет использоваться в большинстве компьютеров NEC PC-9801 и Epson PC.

Устройство имеет стандартный интерфейс SCSI, 256 Кбайт кэш-памяти, хорошие скоростные характеристики и цену в 830 долларов.

Newsbytes, November 9, 1990.

Японский филиал Cray Research объявил о создании двух дисковых устройств для нового суперкомпьютера Cray Y-MP2E.

К одному “дисковому каналу“ накопителей DD-60 и DD-61, работающих под управлением Unix-подобной операционной системы Unicos, можно подсоединить до восьми суперкомпьютеров. В максимальной конфигурации емкость памяти составляет 400 гигабайт. Вообще же максимальный размер файла, поддерживаемый операционной системой, составляет 19 терабайтов.

DD-60 имеет максимальную скорость передачи информации 24 Мбайта в секунду (вдвое больше предыдущих моделей фирмы) и емкость одного устройства 1.96 Гбайт. Привод DD-61 передает данные на скорости до 3 Мбайт в секунду, но может хранить до 2.23 Гбайт информации. Цена DD-60 — 64000 долларов, а DD-61 45400 долларов. Поставки начнутся во втором квартале будущего года.

Newsbytes, November 9, 1990.

Японское подразделение Digital Equipment Corporation создало 12 новых компьютеров линии VAX/VMS. Это 6 новых моделей среднего класса VAX 6000 model 500, две серверных машины, два типа настольных компьютеров общего применения Micro VAX 3100 и два типа серверов к ним.

Все компьютеры поддерживают работу в сетях, содержат центральный процессор, который можно постепенно улучшать, новые платы памяти и усовершенствованные каналы ввода-вывода.

Newsbytes, November 9, 1990.

Фирма Intel впервые реализовала все возможности систем multimedia всего на двух микросхемах. Представитель компании сообщил, что по их мнению эти системы будут встраиваться во все, что содержит микроэлектронику, а не только в персональные компьютеры. Разработанный набор микросхем позволяет работать с большим числом центральных процессоров.

Среди потенциальных применений чипов не только дополнительные платы multimedia для персональных компьютеров, но и улучшенные банковские машины, выдающие деньги, системы торговли и сложные рабочие станции.

По заявлению фирмы, возможно будет использовать новые технологии для упрощения процесса профессионального обучения. Вместо того, чтобы читать в книге, как приготовить салат или откалибровать прибор, обучаемый сможет увидеть реальный процесс и управлять им посредством компьютера.

Образцы видеопроцессора 1750 существуют уже сейчас. Промышленное производство начнется в первом квартале 1991 года. Цена за штуку будет 105 долларов при заказе партии в 1000 штук.

Newsbytes, November 9, 1990.

Advanced Logic Research начала продавать свое новое семейство Compaq Systempro-совместимых машин ALR Powerpro. Самая дешевая машина в серии (7495 долларов) имеет 5 Мбайт ОЗУ (с возможностью расширения до 49 Мбайт), 33-мегагерцевый процессор 80486 с возможностью подключения второго аналогичного процессора. Жесткого диска эта модель не имеет.

Другая модель из новой серии, которая на 30% дешевле аналогичных машин, произведенных Compaq, поставляется в стандартной конфигурации с 17 Мбайтами ОЗУ, 12 слотами расширения, местом для установки до 130 последовательных портов, одним гигабайтом дисковой памяти и двумя микропроцессорами 80486, каждый с 512 Кбайт кэш-памяти. По заявлениям представителей фирмы, производительность машины достигает 40 миллионов инструкций в секунду (MIPS).

Newsbytes, November 9, 1990.

Фирма Globalink заявляет, что ее программа автоматического перевода, стоящая 998 долларов, дешевле человека-переводчика. Программа берет исходный файл на английском или других языках и переводит его со скоростью до 20 тысяч слов в минуту. Но это лишь помощь человеку-переводчику, а не полная замена его.

Программа поддерживает, в зависимости от комплектации, перевод с английского, французского, немецкого, испанского, русского и китайского на любой из этих языков. По заявлению фирмы, реализована также интерпретация идиоматических выражений, контекстный перевод и употребление специальных слов. Программа не только производит подстрочный перевод, но и преобразует структуру предложения в соответствии с требованиями языка.

Ряд пользователей, уже испытавших систему, заявляют об ускорении процесса перевода документации на 80%.

Поставляемый с системой словарь может быть дополнен новыми словами.

В СССР похожими работами занимается группа программистов в Ленинграде. На выставке PC World Forum летом этого года они представляли программу, аналогичную указанной.

Newsbytes, November 14, 1990.

(Окончание. Начало на стр. 25)

Последняя из числа рассматриваемых СУБД — System M занимает особое место среди рассматриваемых систем — это экспериментальное средство обработки транзакций к базе данных, резидентно хранящейся в оперативной памяти. При разработке System M преследовались три цели:

1. Оценить потенциальные преимущества резидентных баз данных. С годами полупроводниковая память становится все дешевле и дешевле. К настоящему времени хранение в оперативной памяти больших массивов информации становится реальностью. За счет этого достигается существенное повышение производительности: сокращается объем ввода-вывода; исключается необходимость кэширования информации, снижаются объемы блокировок данных, используются более эффективные методы поиска и обработки запросов и т.п.

2. Создать архитектуру и алгоритмы, наилучшим образом подходящие для резидентных баз данных. Система обработки транзакций к резидентной базе данных может быть разработана как система, работающая с диском, но имеющая буфер, достаточно большой для хранения всей базы данных. Однако такой подход во многом исключает возможность получения преимуществ, связанных с резидентным хранением данных, поэтому System M разрабатывалась "с нуля" и в предположении возможности резидентного хранения данных, что привело к появлению новой архитектуры внутренней системы обработки и стратегии восстановления.

3. Разработать экспериментальное средство для сравнительной оценки алгоритмов. В частности, в System M реализовано несколько стратегий регистрации и архивирования данных методом контрольных точек.

В настоящее время основное внимание уделяется вопросам восстановления базы данных. Несмотря на то, что System M является средством обработки транзакций, а не программой управления восстановлением данных, она достаточно полно выполняет эти функции. Ряд элементов System M, связанных с восстановлением, реализованы в нескольких вариантах, что позволяет осуществить выбор лучшей альтернативы. Другие элементы System M, — работа в сети, пути доступа и обработка запросов, — пока находятся на достаточно примитивном уровне.

М.Михайлов

На обратной стороне этой страницы помещен бланк заказа на сборник «КомпьютерПресс»

Вы можете его вырезать и, заполнив, отправить в конверте по адресу:

113093, Москва, а/я 37.

В настоящее время принимается подписка на 1991 год. Число экземпляров — без ограничений.

Вы можете выписать журнал на полгода или на год. Стоимость годовой подписки — 48 рублей, полугодовой — 24 рубля.

Деньги следует перечислить на расчетный счет агентства "КомпьютерПресс".

Банковские реквизиты:

получатель: Автобанк (для зачисления на счет №345708)
расчетный счет получателя: №161202
банк получателя: ЦОУ при Госбанке СССР. МФО №299112.
Копию платежного документа необходимо приложить к бланку заказа.

Без одновременной оплаты подписной стоимости заказ не принимается. Издания агентства "КомпьютерПресс" наложенным платежом не высылаются.

В следующем номере “КомпьютерПресс“:

Технофантазии Голливуда и Диснейленда.

В статье речь пойдет о тех сферах жизни, в которых еще недавно использование вычислительной техники было немыслимо. Движущиеся монстры, разрушающиеся небоскребы — все это с помощью компьютеров.

Старый XT с новыми возможностями

Несколько советов пользователю, желающему увеличить производительность XT до уровня AT с процессором i386.

Мультипроцессорные системы

Если дальнейшее увеличение тактовой частоты не может существенно повысить производительность — решение проблемы в создании мультипроцессорной системы.

Компьютер — музыкант, дирижер, аранжировщик

В статье говорится о соединении интеллекта персонального компьютера с грандиозными звуковыми возможностями современных электронных музыкальных инструментов.

ЗАКАЗ

От кого______________________________________

Адрес______________________________________

(ПОЧТОВЫЙ ИНДЕКС УКАЗЫВАТЬ ОБЯЗАТЕЛЬНО)

________________________________тел.____________

Прошу оформить подписку на 1991 год

Подписная плата в сумме________________________перечислена

платежным поручением №_______________от______________199__г.

Официальный партнер в СССР фирмы

NOVELL, INC.

СП "ИНТЕРПРОКОМ"

NetWare 386 Preview: Speedy New Release Was Worth the Wait

PCWeek

Editor's Note: Since early July, Novell Inc.'s NetWare 386 operating system has been beta tested at several sites across the country. Presented on this page and the facing page are the opinions and expe-

By Garry Frenkel

By now most people interested in LANs have heard and read a great deal about Novell's new NetWare 386. Longtime Novell users have eagerly awaited NetWare

lent documentation, to the simple installation (yes, it really does take less than 20 minutes), to the increased functionality, to the near-blinding speed, Novell has done a splendid job.

In our lab at Martin Marietta Data Systems, we have seen applications running on a NetWare 386 server show performance improvements of greater than 40 ... a NetWare 286 server. This is

impressive. The improvements in security are also welcome. Passwords are now encrypted at the workstation before being transmitted to the server.

For users running older versions of the workstation shell, the server can be set to allow unencrypted passwords.

In the past, our monitoring equipment was easily able to read the NetWare passwords as they were transmitted to the

Netware 386 Will Triple Power of Current Version

By Rachel Parker and Mark Stephens

Bringing new firepower to its network operating system war with Microsoft Corp., Novell Inc. will announce its next-generation Netware 386 product today at San Francisco's Palace of Fine Arts, according to sources briefed by the company.

Offering what Novell has said will be at least three times the performance of the company's current top-of-the-line Netware 2.15, Netware 386 is a complete rewrite of the Netware operatin[g] system and runs in protecte[d] mode on 80386-based file serv[ers]. The product is intended t[o] blow the doors off its majo[r] competitor, Microsoft OS/[2] LAN Manager, which runs i[n] 80286 protected mode as a tas[k] under OS/2.

Novell is not expected t[o] abandon its 286-based Netwar[e] 2.1X product line.

Novell president Ray Noor[da] told financial analysts and investors attending the Ham-brecht & Quist ...

InfoWorld

NetWare 386: The network server platform for the '90s

BY JODI MARDESICH

SAN FRANCISCO—The waiting and speculation are over. Novell has unveiled NetWare 386 v3.0 and v3.1, the company's "server platform for the '90s."

"NetWare 386 is a major redesign of the NetWare operating system that takes advantage of 386 architecture," said

NetWare 386 supports up to 250 nodes per server, up to 32GB volumes, with 32 physical drives per volume for a total of 1,024 physical drives per server; 100,000 concurrent open files; more than 2 million directory entries per volume; a maximum ... can span physical

network server operating system."

King said the operating system has been architected in a mod[ular] way, so that users can incrementally ... platform using ser[ver] ... called NetWare Lo[adable Modules] (NLMs).

LAN TIMES

NetWare 386 May Pack Punch To Knock Out the Competition

By Bob Enyart

Novell Inc.'s NetWare 386 won't give Banyan Systems Inc.'s ownership of the premier naming service—StreetTalk—a run for its money, nor will it reduce the unique appeal of Microsoft Corp.'s LAN Manager, with its OS/2 compatibility and such features as automatic disconnect/reconnect.

However, NetWare 386 will blow away the competition—including Novell's own 286-based NetWare 2.15—in performance

func... and flexibility, base... ined...

Novell Brings 'Horsepower' To NetWare

BY TIMOTHY HAIGHT

SAN FRANCISCO — Novell Inc. last week unveiled NetWare 386—its fastest, most extensive, and ultimately, most open local area ...

... also showed off new ... and detailed its strate... network computing

Communications Week

NetWare 386 gets high praise

BY JOEL SHORE
Provo, Utah

Novell Inc., riding a wave of rave reviews from beta testers, last Tuesday began shipping NetWare 386, keeping its word that the "networking platform for the 1990s" would reach users before the end of September.

The first customer to receive a production version of NetWare 386 VINES/386.0 was Coca-Cola Foods of Houston, one of 24 sites that beta-tested the product.

"Corporate America has been growing its LANs faster and pushing them further than anybody could have imagined," said Cheryl Currid, director of Applied Information Technology at Coca-Cola Foods.

Computer Reseller News

Novell wows users with NetWare 386

By Susan Breidenbach
West Coast Bureau Chief

SAN FRANCISCO — As expect[ed], Novell, Inc. took the wraps [off] NetWare 386 here last we[ek] promising 500 attendees of [the] 5th Annual Developers Con[fer]ence that the first release w[ould be] out by early fall.

Designed for Intel 80386-based systems, the version of NetWare is a 32-[bit net]work operating system th[at sup]ports up to 4G bytes of ... able memory, 32 tera...

Network World

NetWare
NetWare
NetWare

ComputerPress

ПРЕДЛАГАЕТ:

к поставке оригинальное сетевое программное обеспечение фирмы NOVELL, INC., оказывает услуги по генерации и установке, проводит обучение и дает консультации по вопросам использования и эксплуатации любых программных продуктов фирмы NOVELL, INC.

Оплата услуг и обучения в рублях!

NOVELL®

119435 Москва, Саввинская наб. 25.
Телефоны: 247-10-31, 245-89-57. Факс: 247-10-52. Телекс 4512249 INPRO SU. Телетайп: 111541 ПУЛЬТ.

Годовая подписка — это экономия Вашего времени!

